TECHNISCHE UNIVERSITÄT BERLIN

Faculty of Electrical Engineering and Computer Science

Master's Thesis

Explaining Anomalies by Taylor-Based Decomposition of Outlier-Ensembles

Zhuo Chen
Registration number: 0395847

13. Februar 2021

First Examiner:	Prof. Dr. Klaus-Robert Müller
Second Examiner:	Prof. Dr. Thomas Wiegand
Supervisor:	Jacob Kauffmann

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle Ausführungen, die anderen veröffentlichten oder nicht veröffentlichten Schriften wörtlich oder sinngemäß entnommen wurden, habe ich kenntlich gemacht.

Die Arbeit hat in gleicher oder ähnlicher Fassung noch keiner anderen Prüfungsbehörde vorgelegen.

Ort, Datum	Unterschrift

Abstract

With the development of neural networks, people have more interest on building models those can be explained with neural networks. In novelty detection task, One-Class deep Taylor decomposition (OC-DTD) has been proved as an efficient method to explain the decision outcome of the One-Class SVM model. In this thesis we extent the OC-DTD model to outlier-ensemble models and propose a method (called "local Times" method) to detect and explain the anomalies. The bagging method is used as the baseline.

The models are validated on MNIST, MNIST-C and real world dataset MVTec. During the experiments the "Clever Hans" effect is observed. Although compared with other methods the "local Times" method has a relative lower accuracy on the classification task, it gives a better explanation for the novelty and suits for practical applications.

Keywords: novelty detection, one-class SVM, explainable machine learning, deep Taylor decomposition.

Zusammenfassung

Mit der Entwicklung neuronaler Netze wächst das Interesse daran, Modelle zu erstellen, die mit neuronalen Netzen erklärt werden können. In der Ausreißererkennung hat sich die One-Class Deep Taylor Decomposition (OC-DTD) als eine effiziente Methode zur Erklärung des Entscheidungsergebnisses von One-Class SVM Modellen erwiesen. In dieser Arbeit erweitern wir das OC-DTD Modell auf Ausreißer-Ensemble-Modelle und schlagen die sogenannte "lokale Multiplikationsmethode" vor, um die Abnormität zu erkennen und zu erklären. Die Bagging Methode wird als das Baseline-Modell verwendet.

Die Modelle werden auf MNIST, MNIST-C und dem realen Datensatz MVTec validiert. In den Experimenten wird der "Kluge Hans"-Effekt beobachtet. Obwohl die "lokale Multiplikationsmethode" im Vergleich zu anderen Methoden eine relativ geringere Genauigkeit bei der Klassifizierung aufweist, bietet es eine bessere Erklärung für die Abnormität und ist für praktische Anwendungen geeignet.

Schlüsselwörte: Ausreißererkennung, One-Class SVM, erklärbares maschinelles Lernen, Deep Taylor Decomposition

Contents

Chapter 1

Introduction

Anomaly detection[1, 2] is very close to people's lives in this era of information[3, 4]. Compared with the normal data, people usually have more concern on the abnormal data, as the outliers may indicate the potential threats, e.g. a nodule from CT images or an intrusion in computer systems.

Novelty detection is a special method among various of algorithms in anomaly detection, the model is only trained on the clean data (i.e. all the training samples are inliers) while it can be used to analysis whether a test sample is an outlier or not. In practical, One-Class SVM[5] has been widely applied in novelty detection as a well-studied algorithm[6, 7]. However, there are limitations to this method: even though we build a classifier, which can classify the inliers and outliers with 100 percents accuracy, this model cannot help to explain why a test sample should be predicted as outlier or which parts of this test sample is anomalous. Towards finding a suitable explanation for outlier detection, Kauffmann et al.[8] proposed a new method (called "OC-DTD") in 2018 , which is a combinition of One-Class SVM and Deep Taylor Decomposition (DTD). Based on One-Class SVM, a function in OC-DTD is established to quantify the outlierness of data. After building the corresponding neural network for this outlierness function, by applying Deep Taylor Decomposition the relevance [9] score of outlierness can be redistributed onto the input layer, i.e. each dimension of the input sample. With the help of heatmap[10], OC-DTD achieves the visualization of outliers detection in digital image processing, in other words, the highlight parts of the heatmap indicate the anomaly.

Ensemble learning[11] is a common technique being used to improve the performance of the models. In this thesis, we test several outlier-ensemble models, which are all generated from two single OC-DTD outlierness models. We also build different neural networks for each ensemble model respectively, and test the models on three different datasets, i.e. the simplest clean dataset MNIST, the noisy dataset MNIST-C[12] and the high resolution images dataset MVTec[13]. The experiment results show that one of these models and its neural network structure (called "local Times" method) has a better explanation for the outliers than the baseline method "bagging"[14], although it has relative lower accuracy on the classification task.

Related work

1.1. One-Class SVM and Outlierness function

One-Class SVM is proposed by Schölkopf et al.[5] and it is an efficient method for novelty detection. Assume there are training data $x_1, ..., x_n \in \chi$ and feature map $\phi : \chi \to \mathcal{F}$, the primal One-Class SVM task is:

$$\begin{aligned} &\min_{w \in, \xi \in R^n, \rho \in R} \quad \frac{1}{2}||w||^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho \\ &subject\ to \quad \forall i = 1, ..., n, \\ &\langle w, \phi(x_i)\rangle \geq \rho - \xi_i, \\ &\xi_i \geq 0 \end{aligned} \tag{1.1}$$

where $\nu \in (0, 1)$ is a lower bound for the fraction of samples that are support vectors(SVs) and an upper bound for the fraction of samples that are on the wrong side of the hyperplane. After solving this quadratic program we can build the decision function:

$$f(x) = sgn\left(\sum_i \alpha_i k(x, u_i) - \rho\right) \tag{1.2}$$

where u_i are SVs and $k(x, u_i) = \langle\phi(x), \phi(u_i)\rangle$.

For any test data x, if $f(x) < 0$, then x would be classified as an outlier, otherwise as an inlier. In this thesis we only discuss the most common Gaussian kernel, i.e. $k(x, y) = \exp\left(-\gamma \left\|x - y\right\|^2\right)$. The solution of OC-SVM can be explained as mapping the data into the feature space corresponding to the kernel and finding a hyperplane with maximum margin from the origin, which satisfies that the inliers and outliers fall on different sides of this hyperplane. The OC-SVM model can classify the in- and outliers well, but it has the limit that it cannot quantify the outlierness of data points.

From equation (1.2) we know that the smaller $\sum_i \alpha_i k(x, u_i) - \rho$ is, the higher probability the data would be predicted as outlier. Thus, this term can be used to build a function that quantifies outlierness. To achive this goal Kauffmann et al.[8] established a function $o(x)$ based on the decision function of OC-SVM.

$$
\begin{aligned}
o(x) &= -\log\left[\sum_i \alpha_i k(x, u_i)\right] \\
&= -\log\left[\sum_i \exp\left(-\gamma \left\|x - u_i\right\|^2 + \log \alpha_i\right)\right]
\end{aligned}
\tag{1.3}
$$

This outlierness function $o(x)$ fulfills three conditions:

1. It is lower bounded by zero;

2. It converges asymptotically with some predefined norm, i.e. $\forall x \neq 0$, $\lim_{t \to +\infty} o(tx)/||tx||^q = c$, for some $q, c > 0$;

3. The higher value means the higher outlierness.

1.2. Deep Taylor Decomposition and Heatmap

Deep Taylor decomposition is a method that decomposes the output of a neural network onto the input variables[9, 15] and has been widely applied in explainable machine learning [16, 17, 18, 19]. In DTD a new notion "relevance" is proposed, the relevance score of a neuron indicates how much the neuron is relevant for the prediction outcome.

Assume layer A and layer B are two layers in neural network that connects with each other, A is the previous layer of B, $\{a_i\}$,$\{b_j\}$ are the sets of all the neurons in layers A and B respectively. In this thesis we use the following definitions.

$R_{\{a_i\}}, R_{\{b_j\}}$: the relevance scores of neurons a_i and b_j. $R_{\{b_j\}}$ is a function of neuron set $\{a_i\}$;

$R_{\{o\}} = o$: the relevance score of output layer, which is defined as the outcome of the neural network;

$R_{\{b_j \to a_i\}}$: the relevance score of neuron b_j to neuron a_i.

The relationship between these relevance scores and layer-wise Taylor decomposition can be described with the formulas below:

$$
R_{\{b_j\}} = R_{\{b_j\}}(\tilde{a}) + \nabla R_{\{b_j\}}(\tilde{a})^T (a - \tilde{a}) + \epsilon \tag{1.4}
$$

$$
R_{\{b_j \to a_i\}} := \nabla_i R_{\{b_j\}}(\tilde{a}) \cdot (a_i - \tilde{a}_i) \tag{1.5}
$$

$$
R_{\{a_i\}} := \sum_j R_{\{b_j \to a_i\}} \tag{1.6}
$$

Where $\tilde{a}$ is a root point that satisfies $R_{\{b_j\}}(\tilde{a}) = 0$; ϵ contains all high order terms and satisfies $\epsilon \approx 0$ at $\tilde{a}$. Moreover, if $\epsilon = 0$, then

$$
R_{\{b_j\}} = 0 + \nabla R_{\{b_j\}}(\tilde{a})^T (a - \tilde{a}) + 0 = \sum_i R_{\{b_j \to a_i\}} \tag{1.7}
$$

$$\sum_j R_{\{b_j\}} = \sum_j \sum_i R_{\{b_j \to a_i\}} = \sum_i \sum_j R_{\{b_j \to a_i\}} = \sum_i R_{\{a_i\}} \tag{1.8}$$

When equation (1.8) holds, we call the layer-wise Taylor decomposition is conservative. If each layer-wise Taylor decomposition is conservative, then the sum of the relevance scores in any single layer l is always identical to the sum in the output layer. For the single output neural network, we have $\sum_i R_{\{l_i\}} = R_{\{o\}} = o$.

The idea of DTD is continuously redistributing the relevance scores of neurons to each neuron in previous layer by applying layer-wise Taylor decomposition. Then $R_{\{o\}}$ is redistributed to the input neurons $\{x_i\}$, the relevance scores $R_{\{x_i\}}$ can be used to generate the heatmap, which visualizes the contributions of the input data to the prediction outcome.

1.3. One-Class Deep Taylor Decomposition

One-class deep Taylor decomposition (OC-DTD) is a method that explains the outlierness function of OC-SVM with neural network. It is usually applied in detecting the anomaly in image data, the relevance-based heatmap shows which pixels of the image are important for the prediction.

1.3.1. Neural network structure of One-Class SVM

For Gaussian kernel, the outlierness function of One-Class SVM can be expressed as a three-layer neural network: (1) The input layer x; (2) the hidden layer $h_i(d) = \gamma d_i^2 - \log \alpha_i$, where $d_i(x) = \|x - u_i\|$; (3) the pooling layer $o(h) = -\log[\sum_i \exp(-h_i)]$. The network structure is shown in Fig 1.1.

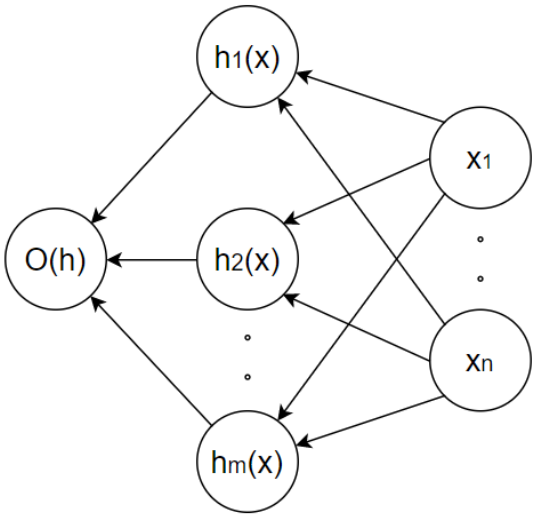

Figure 1.1. neural network for One-Class SVM

1.3.2. Relevance of the input layer

Our goal is to redistribute the relevance of outlierness on the input layer x. The layer-wise relevance propagation for the OC-SVM with neural network like Fig 1.1 can be expressed with Fig 1.2.

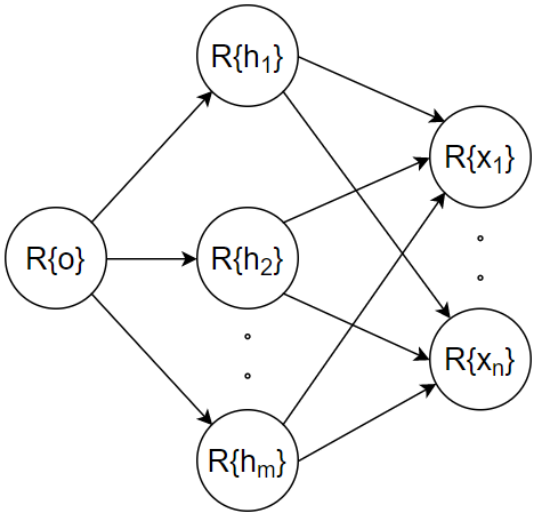

Figure 1.2. Layer-wise relevance propagation for One-Class SVM

First, we redistribute $R_{\{o\}} = o$ (i.e. the relevance of the output) on the hidden layer h by applying Taylor decomposition. As $o(h) = -\log[\sum_i \exp(-h_i)]$, we choose $\tilde{h} = h - o(h)$ as the root point, which satisfies $R_{\{o\}}(\tilde{h}) = 0$ and $\epsilon = 0$ (see proof in Appendix A.1. and A.2.). The $R_{\{h_i\}}$ can be computed as below,

$$R_{\{h_i\}} = \nabla_i R_{\{o\}}(\tilde{h}) \cdot (h_i - \tilde{h}_i) = \frac{\exp(-h_i)}{\sum\limits_k \exp(-h_k)} \cdot o \tag{1.9}$$

In the next step, we redistribute $R_{\{h_i\}}$ on the input layer x. But $R_{\{h_i\}}$ is complex to analyze, as the result of that, in DTD we replace the $R_{\{h_i\}}$ by another relevance function $\tilde{R}_{\{h_i\}}$, which satisfies $\tilde{R}_{\{h_i\}} \approx R_{\{h_i\}}$ at the current activation h and in its vicinity [9].

Let $w_i = \frac{\exp(-h_i)}{\sum\limits_k \exp(-h_k)}$ and $\epsilon_i = o - h_i$. Then one can show asymptotic constancy of w_i and ϵ_i when i dominates the min-pooling (see proof in Appendix A.3.). Hence, we have,

$$\begin{aligned} \tilde{R}_{\{h_i\}} \approx R_{\{h_i\}} &:= w_i(h_i + \epsilon_i) \\ &= w_i\gamma||x - u_i||^2 + w_i(\epsilon - \log\alpha_i) \\ &= C_i||x - u_i||^2 + D_i \end{aligned} \tag{1.10}$$

where $C_i = w_i\gamma > 0$, $D = w_i(\epsilon - \log\alpha_i)$ are treated as constants.

By applying the Integrated Gradients[20] and choosing the segment from u_j to x as the integrated route, let $u_i^{(j)}$ denote the j-th element of vector u_i, the relevance $R_{\{h_i \to x_j\}}$ can be expressed as below,

$$R_{\{h_i \to x_j\}} = \frac{(x_j - u_i^{(j)})^2}{||x - u_i||^2}(R_{\{h_i\}} - D_i) \tag{1.11}$$

$$
\begin{aligned}
R_{\{x_j\}} &= \sum_i R_{\{h_i \to x_j\}} \\
&= \sum_i \frac{(x_j - u_i^{(j)})^2}{||x - u_i||^2}(R_{\{h_i\}} - D_i)
\end{aligned}
\tag{1.12}
$$

We find that, when $D_i < 0$, then $\sum_j R_{h_i \to x_j} = R_{\{h_i\}} - D_i > R_{\{h_i\}}$, which means extra relevance is added to the model. To avoid this situation, we replace D_i by $max\{0, D_i\}$. Thus, the relevance score of input layer in OC-DTD can be expressed as below,

$$
R_{\{x_j\}} = \sum_i \frac{(x_j - u_i^{(j)})^2}{||x - u_i||^2}(R_{\{h_i\}} - max\{0, D_i\}) \tag{1.13}
$$

Moreover, we could use Form (1.14) to implement the OC-DTD,

$$
R_{\{x\}} = (\frac{\partial d}{\partial x})^2 \cdot [min(o, \frac{d}{2} \odot \frac{\partial h}{\partial d}) \odot \frac{\partial o}{\partial h}] \tag{1.14}
$$

where $d = (\|x - u_j\|)_j$ is the vector of distances between x and the support vectors.

1.4. Pixel-Flipping Evaluation Metric

Pixel-Flipping is a metric being used to evaluate the heatmaps of samples[21]. The idea of Pixel-Flipping is masking the pixels (i.e. reducing the data dimensions) form the highest to the lowest relevance score, and at the same time measuring how fast the prediction score decreases.

Assume there are OC-SVM model C and one data point $x \in R^d$, the outlierness score of x is $o(x) = -\log\left[\sum_{i=1}^{n} \alpha_i k(x, u_i)\right]$. By applying the form (1.14) we obtain the relevance score $R_{\{x\}}$. We use $s_1, s_2, ..., s_d$ to denote the sorted sequence of numbers $1, 2, ..., d$, which satisfies $R_{\{x^{(s_1)}\}} \geq R_{\{x^{(s_2)}\}} \geq ... \geq R_{\{x^{(s_d)}\}}$, then the outlierness function can be rewritten as below:

$$
\begin{aligned}
o(x) &= -\log\left[\sum_{i=1}^{n} \alpha_i k(x, u_i)\right] \\
&= -\log\left[\sum_{i=1}^{n} \exp\left(-\gamma \left\|x - u_i\right\|^2 + \log \alpha_i\right)\right] \\
&= -\log\left[\sum_{i=1}^{n} \exp\left(\log \alpha_i - \gamma \sum_{j=1}^{d} (x^{(j)} - u_i^{(j)})^2\right)\right] \\
&= -\log\left[\sum_{i=1}^{n} \exp\left(\log \alpha_i - \gamma \sum_{j=1}^{d} (x^{(s_j)} - u_i^{(s_j)})^2\right)\right]
\end{aligned}
\tag{1.15}
$$

In approach of Pixel-Flipping we mask k input pixels with highest relevance score, i.e. pixels $x^{(s_1)}, ..., x^{(s_k)}$, then the dimension of x is reduced from d to $d - k$. Let $o_k(x)$ be the outlierness function of the masked data, it is defined as the following:

$$o_k(x) = -\log\left[\sum_{i=1}^{n}\exp\left(\log\alpha_i - \gamma\sum_{j=k+1}^{d}(x^{(s_j)} - u_i^{(s_j)})^2\right)\right] \tag{1.16}$$

Based on Form (1.16) we can deduce the properties of $o_k(x)$: 1) $o_k(x)$ would decrease when k grows; 2) $o_0(x) = o(x)$, i.e. $o(x)$ is the special situation that none of the pixels is masked; 3) o_d(x) = 0, i.e. when all the pixels are masked, the outlierness score should be zero.

At last, we introduce a new variable $ratio(k)$, which is defined as the fraction between $o_k(x)$ and $o(x)$. When k increase from 0 to d, we can calculate the corresponding $ratio(k)$ values and draw a curve (called PF curve) for $ratio(k)$. This curve indicates how fast the outlierness converge to zero and is used to evaluate the explanation for x. If the PF curve decreases fast when k grows, it means the heatmap has captured the most relevant pixels for the prediction score (i.e. the outlierness), in other words, this heatmap and the outlierness function lead to a good explanation. The faster the PF curve decreases, the better the explanation is.

Chapter 2

Ensemble Outlierness Score

Ensemble techniques have been widely applied in machine learning[22, 23]. In most situations the performance of ensemble models are better than the basic single model. This thesis focus on analysing the decomposition of the ensembles of outlierness functions.

Assume there are two groups of parameters of One-Class SVM models C_1 and C_2:

$$\begin{aligned} o_1(x) &= -\log\left(\sum_{i=1}^{m} a_i \exp(-\gamma\|x-u_i\|^2)\right) \\ o_2(x) &= -\log\left(\sum_{j=1}^{n} b_j \exp(-\gamma\|x-u'_j\|^2)\right) \end{aligned} \tag{2.1}$$

where u_i and u'_j are support vectors, a_i and b_j are the corresponding Lagrange multiplier. We use two methods (called "Bagging" and "Times" methods) to fuse the models C_1 and C_2, where the ensemble outlierness functions are built based on o_1 and o_2. Then we decompose these functions with different neural network structures and analysis their results.

2.1. Bagging method

The bagging method has been proved as an efficient ensemble technique that can improve the robustness of the classifiers[24, 25]. By applying bagging, the ensemble outlierness function is chosen as the mean of the outlierness functions of C_1 and C_2.

$$\begin{aligned} o_p(x) &= \frac{1}{2}\left(o_1(x)+o_2(x)\right) \\ &= -\frac{1}{2}\log\left(\sum_{i=1}^{m} a_i \exp(-\gamma\|x-u_i\|^2)\right) \\ &\quad -\frac{1}{2}\log\left(\sum_{j=1}^{n} b_j \exp(-\gamma\|x-u'_j\|^2)\right) \end{aligned} \tag{2.2}$$

As the number of support vectors in equation (2.2) is $m+n$ in total (where m and n are the numbers of support vectors in C_1 and C_2 respectively), we use o_p to denote the ensemble outlierness function.

After building the new outlierness function, we use two neural network structures to explain it but find out they share an identical result.

2.1.1. Neural network 1

As $o(x)$ is the mean of $o_1(x)$ and $o_2(x)$, the simplest neural network is adding a mean-pooling layer to the network for OC-SVM (Fig 1.1) . The layer-wise relevance propagation for network 1 can be expressed as Fig 2.1.

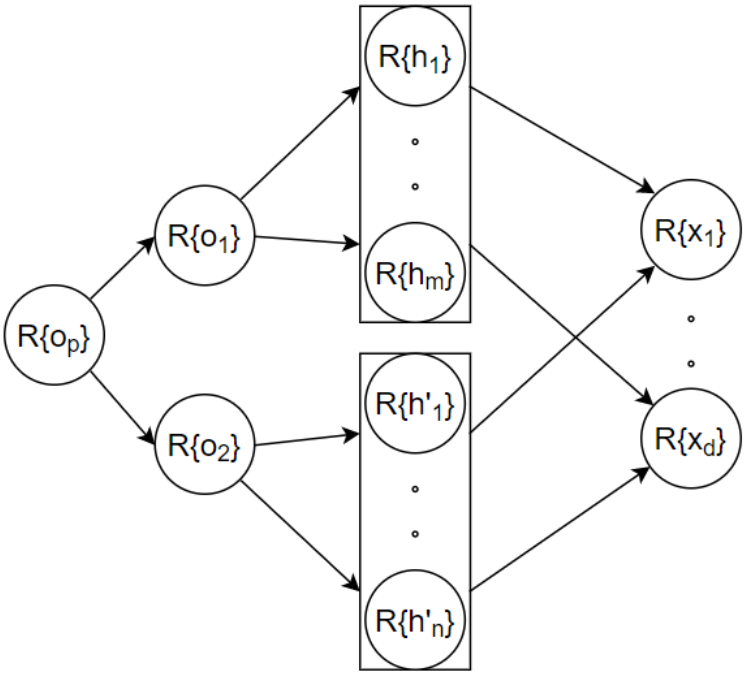

Figure 2.1. Layer-wise relevance propagation for neural network 1

At first, we decompose $R_{\{o_p\}}$ into $R_{\{o_1\}}$ and $R_{\{o_2\}}$, by applying Taylor decomposition we choose $\tilde{o}_1 = \tilde{o}_2 = 0$ as the root point and obtain $R_{\{o_1\}} = \frac{1}{2}o_1$, $R_{\{o_2\}} = \frac{1}{2}o_2$; Then we redistribute $R_{\{o_1\}}$ and $R_{\{o_2\}}$ onto input layer separately by applying equation (1.14); Finally, we sum up the redistributed relevance that flows form $R_{\{o_1\}}$ and $R_{\{o_2\}}$. Thus, the relevance score of the input can be expressed as below:

$$\begin{aligned} R_{\{x\}} = &\frac{1}{2}(\frac{\partial d}{\partial x})^2 \cdot [min(o_1, \frac{d}{2} \odot \frac{\partial h}{\partial d}) \odot \frac{\partial o_1}{\partial h}] \\ &+ \frac{1}{2}(\frac{\partial d'}{\partial x})^2 \cdot [min(o_2, \frac{d'}{2} \odot \frac{\partial h'}{\partial d'}) \odot \frac{\partial o_2}{\partial h'}] \end{aligned} \tag{2.3}$$

2.1.2. Neural network 2

Let $H_{ij} = h_i + h'_j$, we rewrite the ensemble outlierness function (2.2) as below:

$$\begin{aligned} o_p(x) &= \frac{1}{2}(o_1(x) + o_2(x)) \\ &= -\frac{1}{2}\left[\log\left(\sum_{i=1}^{m}\exp(-h_i)\right) + \log\left(\sum_{j=1}^{n}\exp(-h'_j)\right)\right] \\ &= -\frac{1}{2}\log\left(\sum_{i,j}\exp(-h_i - h'_j)\right) \\ &= -\frac{1}{2}\log\left(\sum_{i,j}\exp(-H_{ij})\right) \end{aligned} \tag{2.4}$$

Based on equation (2.4) and Fig 2.1, we can identify a different network structure. We replace the layer of o_1 and o_2 in neural network 1 with a new layer H, which consists of a group of neurons $H_{ij} = h_i + h'_j$. Then we set the pooling layer as $o(H) = -\log[\sum_i \exp(-H_{ij})]$. Thus, The layer-wise relevance propagation for network 2 can be expressed as Fig 2.2.

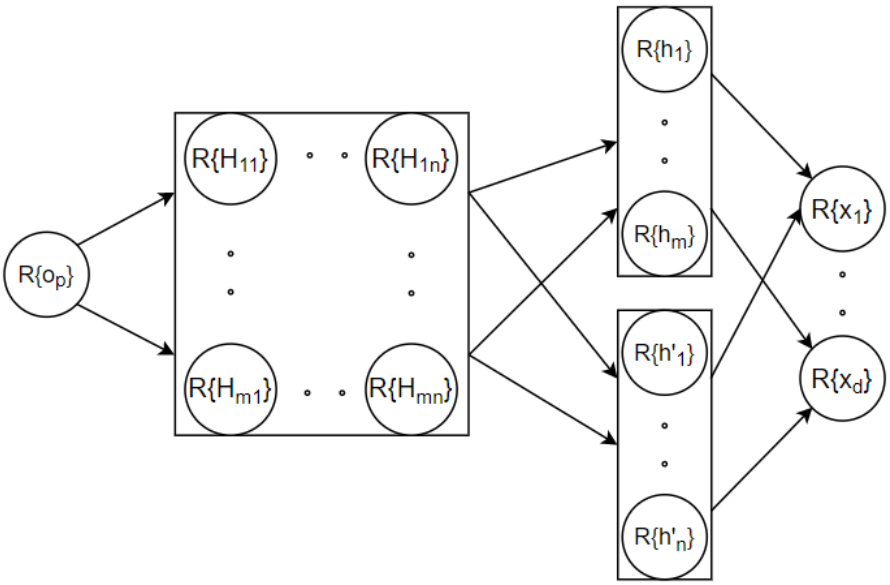

Figure 2.2. The layer-wise relevance propagation for network 2

Now we decompose $R_{\{o_p\}} = o_p(H)$ onto layer H. Assume $\tilde{h} = h - o_1$ and $\tilde{h}' = h' - o_2$, we choose $\tilde{H}_{ij} = \tilde{h}_i + \tilde{h}'_j$ as the root point for deep Taylor decomposition, as it satisfies the conditions $o_p(\tilde{H}) = \frac{1}{2}o_1(\tilde{h}) + \frac{1}{2}o_2(\tilde{h}') = 0 + 0 = 0$ and the high order terms ϵ sum to zero. By applying first order Taylor decomposition we can compute the

relevance score of neuron H_{ij}:

$$\begin{aligned} R_{\{H_{ij}\}} &= \left.\frac{\partial R_{\{o_p\}}}{\partial H_{ij}}\right|_{H=\tilde{H}} \cdot (H_{ij} - \tilde{H}_{ij}) \\ &= \frac{\exp(-H_{ij})}{\sum\limits_{s,t}\exp(-H_{st})} \cdot o_p \\ &= \frac{\exp(-h_i)}{\sum\limits_{s}\exp(-h_s)} \cdot \frac{\exp(-h'_j)}{\sum\limits_{t}\exp(-h'_t)} \cdot o_p \end{aligned} \tag{2.5}$$

Then we redistribute $R_{\{H_{ij}\}}$ on the neurons of previous layer, i.e. h and h'. As $\tilde{h} = h - o_1$ and $\tilde{h}' = h' - o_2$ are also the root point of $R_{\{H_{ij}\}}$, we apply first order Taylor decomposition again and decompose it on neurons h_i and h'_j. The result shows that the $R_{\{h_i\}}$ and $R_{\{h'_j\}}$ are same as in neural network 1 (see Appendix A.4. for a proof). After that, we redistribute $R_{\{h\}}$ and $R_{\{h'\}}$ onto input layer x. We notice that, neural network 1 and neural network 2 share the same structure between the hidden layer h, h' and input layer x. Therefore, the relevance scores $R_{\{x\}}$ should also be identical.

2.2. "Times" Method

In this section we start with analyzing the ensemble outlierness function $o_p(x)$. Multiply both sides of the equation (2.2) by 2, then function $2o_p(x) = o_1(x) + o_2(x)$ can be rewritten as below:

$$\begin{aligned} 2o_p(x) &= o_1(x) + o_2(x) \\ &= -\log\left(\sum_{i=1}^{m} a_i \exp(-\gamma \left\|x - u_i\right\|^2)\right) - \log\left(\sum_{j=1}^{n} b_j \exp(-\gamma \left\|x - u'_j\right\|^2)\right) \\ &= -\log\left(\sum_{i=1}^{m}\sum_{j=1}^{n} a_i b_j \exp(-\gamma \left\|x - u_i\right\|^2 - \gamma \left\|x - u'_j\right\|^2)\right) \\ &= -\log\left(\sum_{i,j} a_i b_j \exp(-2\gamma \left\|x - \frac{1}{2}(u_i + u_j)\right\|^2) - \frac{\gamma}{2}\left\|u_i - u'_j\right\|^2\right) \\ &= -\log\left(\sum_{i,j} a_i b_j \exp(-\frac{\gamma}{2}\left\|u_i - u'_j\right\|^2) \cdot \exp(-2\gamma \left\|x - \frac{1}{2}(u_i + u'_j)\right\|^2)\right) \end{aligned} \tag{2.6}$$

let $\alpha_{ij} = a_i b_j \exp\left(-\frac{\gamma}{2}\left\|u_i - u'_j\right\|^2\right)$, $\alpha = \sum_{i,j}\alpha_{ij}$, $\beta_{ij} = \frac{\alpha_{ij}}{\alpha}$, $\gamma' = 2\gamma$, $u_{ij} =$

$\frac{1}{2}(u_i + u'_j)$, then it can be written as,

$$\begin{aligned} 2o_p(x) &= -\log\left(\sum_{i,j} \alpha_{ij} \exp\left(-\gamma' \|x - u_{ij}\|^2\right)\right) \\ &= -\log\left(\alpha \cdot \sum_{i,j} \frac{\alpha_{ij}}{\alpha} \exp\left(-\gamma' \|x - u_{ij}\|^2\right)\right) \\ &= -\log\left(\sum_{i,j} \beta_{ij} \exp\left(-\gamma' \|x - u_{ij}\|^2\right)\right) - \log(\alpha) \end{aligned} \tag{2.7}$$

The fomula (2.7) shows that the sum of two outlierness function can also be expressed as the outlierness of a single One-Class SVM, in which u_{ij} and β_{ij} are the corresponding support vectors and Lagrange multipliers, $-\log\alpha$ is constant bias. Remove this constant bias, we build the outlierness function of "Times" method like below:

$$\begin{aligned} o_t(x) &= 2o_p(x) - (-\log(\alpha)) \\ &= -\log\left(\sum_{i,j} \beta_{ij} \exp\left(-\gamma' \|x - u_{ij}\|^2\right)\right) \end{aligned} \tag{2.8}$$

This composition is identical to the network from Fig 1.2. As the number of support vectors in equation (2.8) is $m * n$ (where m and n are the numbers of support vectors in C_1 and C_2 respectively), this method is called "Times" method. The equation (2.8) shows that there is a positive linear relationship between o_p and o_t, which means, if we use o_p and o_t to classify the samples, the ROC_AUC scores would be the same.

Depends on the convexity of ground truth decision space, we build different neural networks for o_t by applying OC-DTD.

2.2.1. Neural network 3

We know that OC-SVM is trained on the set of inliers, the support vectors of OC-SVM are the train data points on the decision boundary. Although decision space of OC-SVM may not match the ground truth decision space, the support vectors of OC-SVM must be inliers. Thus, the support vectors u_i, u'_j of well-trained OC-SVM C_1 and C_2 should all locate in the ground truth decision space.

When the ground truth decision space is globally convex, then for any two points A, B in this convex space, the middle point of segment AB should lie in this space too. Hence, all the points $u_{ij} = \frac{1}{2}(u_i + u'_j)$ are in the ground truth space. In this situation, we call this method as "global Times" method.

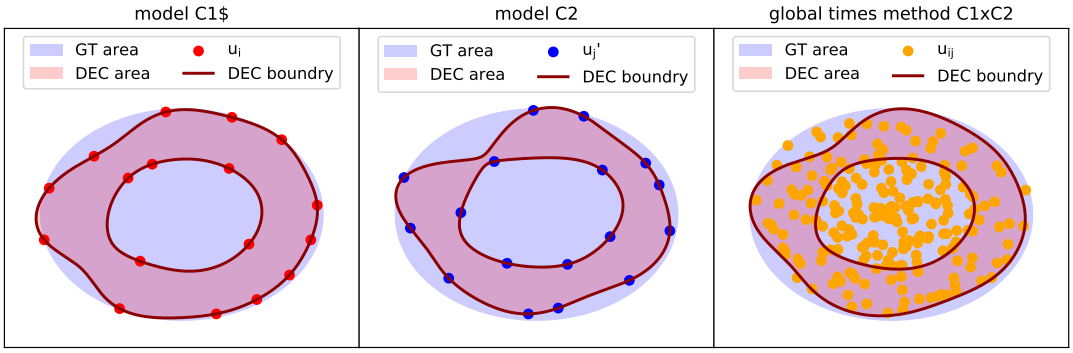

Figure 2.3. Distribution of support vectors of different classifiers in convex ground truth space

The idea of OC-DTD is to explain the sample towards the nearby support vector(s). Although o_p and o_t lead to same ROC_AUC score, the explanation of a same sample would be different. From Fig 2.3 we know that, the distribution of u_{ij} is much more dense in ground truth decision space than u_i and u'_j. As result of that, for most of data points, especially for the inliers, it will be more possible to find u_{ij} rather than u_i or u'_j nearby. Besides, when the distance between u_i and u'_j are too large, β_{ij} goes to zero, i.e. the "effective distance" $h_{ij} = \gamma'||x - u_{ij}||^2 - \log(\beta_{ij})$ becomes very large, even though x is close to u_{ij}. Which means the model can filter this type of middle points and avoid explaining the predict outcome towards them. Thus, for most of the test samples, the "Times" method would give an better explanation.

When C_1 and C_2 are two different classifiers, then there are mn different u_{ij}. Let $D_{ij} = \|x - u_{ij}\|$, $H_{ij} = \gamma' D_{ij}^2 - \log(\beta_{ij})$ and $o_t = -\log\big(\sum_{i,j} \exp(-H_{ij})\big)$. We build neural network 3 based on the structure in Fig 1.1. The hidden layers h and d are replaced with H and D respectively. Thus, the deep Taylor decomposition of o_t is like in Fig 2.4.

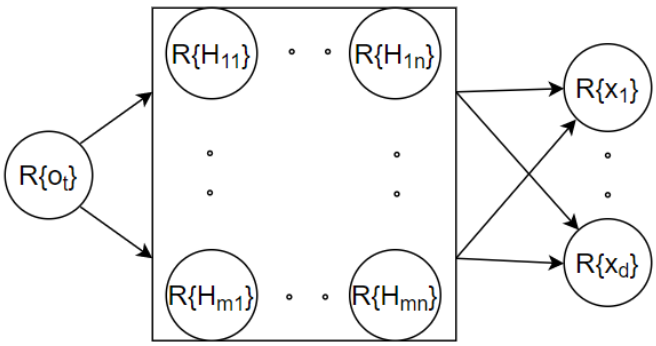

Figure 2.4. The layer-wise relevance propagation for network 3, when C_1 and C_2 are different

When $C_1(x)$ and $C_2(x)$ are two absolutely same OC-SVM, i.e. we fuse two same classifiers in "Times" method. Then there are constraints $u_{ij} = u_{ji}$ and $\beta_{ij} = \beta_{ji}$ in formula (2.8). Hence, there are only $\frac{1}{2}m(m+1)$ different support vectors in formula (2.8).

Let $\forall j \leq i, D_{ij} = \|x - u_{ij}\|$, $H_{ij} = \gamma' D_{i,j}^2 - \log(2\beta_{ij})$, $H_{ii} = \gamma' D_{ii}^2 - \log(\beta_{ii})$ and $o_t = -\log\big(\sum_{j\leq i} \exp(-H_{ij})\big)$. In this situation, the new layer-wise relevance propagation is shown in Fig 2.5.

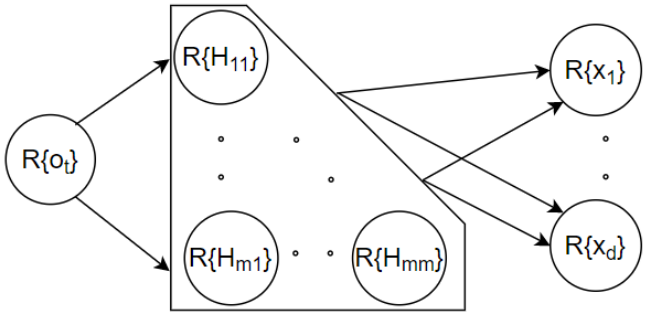

Figure 2.5. The layer-wise relevance propagation for network 3, when C_1 and C_2 are identical

According to Fig 1.1, Fig 2.4, Fig 2.5 and equation (1.14), no mater C_1 and C_2 are different or not, the relevance score of input layer is:

$$R_{\{x\}} = (\frac{\partial D}{\partial x})^2 \cdot [\min(o_t, \frac{D}{2} \odot \frac{\partial H}{\partial D}) \odot \frac{\partial o_t}{\partial H}] \tag{2.9}$$

2.2.2. Neural network 4

In this section we assume the ground truth decision space is not global convex any more. Although u_i and u'_j are still in the ground truth decision space, it can only ensure the local convexity around u_i and u'_j while the middle points $\frac{1}{2}(u_i + u'_j)$ may fall out of the ground truth decision space. If $\frac{1}{2}(u_i + u'_j)$ becomes an outlier, it will be not proper to explain the test samples towards this outlier. Therefore, We avoid using u_{ij} those are outliers to build the neural network and introduce the "local Times" method.

With the same definition of variables in equation (2.6), We rewrite equation (2.8) as below:

$$\begin{aligned} o_t(x) &= -\log\left(\sum_{i,j} \beta_{ij} \exp\left(-\gamma' \|x - u_{ij}\|^2\right)\right) \\ &= -\log\left[\sum_i \left(\sum_j \beta_{ij} \exp\left(-\gamma' \|x - u_{ij}\|^2\right)\right)\right] \end{aligned} \tag{2.10}$$

The relationship between the dual coefficients $\forall \beta_{rp}, \beta_{rq} \in \{\beta_{ij}\}$ is constrained by:

$$\frac{\beta_{rp}}{\beta_{rq}} = \frac{b_p}{b_q} \exp(\frac{\gamma}{2} \left\|u_r - u'_q\right\|^2 - \frac{\gamma}{2} \left\|u_r - u'_p\right\|^2) \tag{2.11}$$

when $\left\|u_r - u'_p\right\| \gg \left\|u_r - u'_q\right\|$, then $\beta_{rp}/\beta_{rq} \to 0$, which means, compared with the weight of support vector u_{rq}, the weight of support vector u_{rp} could be ignored (i.e. set to zero). Besides, as the ground truth decision space is not global convex but local convex around u_i or u'_j, if we only keep the part of u_{ij}, which are near u_i or u'_j, ignore the rest middle points, it will reduce the probability that the middle point u_{ij} falls out of the ground truth decision space.

We build a new outlierness function o'_t which approximately equals to o_t. We define several new variables as following:

k, l: two constants s.t. $1 \leq k \leq m$, $1 \leq l \leq n$;

$v'_{is}, s \in \{1, ..., k\}$: the k-nearest neighbors of u_i in set $\{u'_j\}$;

$v_{tj}, t \in \{1, ..., l\}$: the l-nearest neighbors of u'_j in set $\{u_i\}$;

$w_{is} := \frac{1}{2}(u_i + v'_{is})$;

$w'_{tj} := \frac{1}{2}(u'_j + v_{tj})$.

Based on equation (2.10) we have:

$$\begin{aligned} o_{t1} :&= -\log\left(\sum_{i=1}^{m}\sum_{s=1}^{k}\beta_{is}\exp\left(-\gamma'\left\|x - w_{is}\right\|^2\right)\right) \\ &\approx -\log\left[\sum_{i=1}^{m}\left(\sum_{j=1}^{n}\beta_{ij}\exp\left(-\gamma'\left\|x - u_{ij}\right\|^2\right)\right)\right] \\ &= o_t \end{aligned} \tag{2.12}$$

similarly,

$$\begin{aligned} o_{t2} :&= -\log\left(\sum_{j=1}^{n}\sum_{t=1}^{l}\beta_{tj}\exp\left(-\gamma'\left\|x - w'_{tj}\right\|^2\right)\right) \\ &\approx -\log\left[\sum_{j=1}^{n}\left(\sum_{i=1}^{m}\beta_{ij}\exp\left(-\gamma'\left\|x - u_{ij}\right\|^2\right)\right)\right] \\ &= o_t \end{aligned} \tag{2.13}$$

then we build o'_t based on (2.12) and (2.13), the distributions of u_{ij}, w_{is}, w'_{tj} are shown in Fig 2.6. The layer-wise relevance propagation of "local Times" method is shown in Fig 2.7.

$$\begin{aligned} o'_t :&= -\frac{1}{2}\log\left(\sum_{i=1}^{m}\sum_{s=1}^{k}\beta_{is}\exp\left(-\gamma'\left\|x - w_{is}\right\|^2\right)\right) \\ &\quad -\frac{1}{2}\log\left(\sum_{j=1}^{n}\sum_{t=1}^{l}\beta_{tj}\exp\left(-\gamma'\left\|x - w'_{tj}\right\|^2\right)\right) \\ &= \frac{1}{2}(o_{t1} + o_{t2}) \approx o_t \end{aligned} \tag{2.14}$$

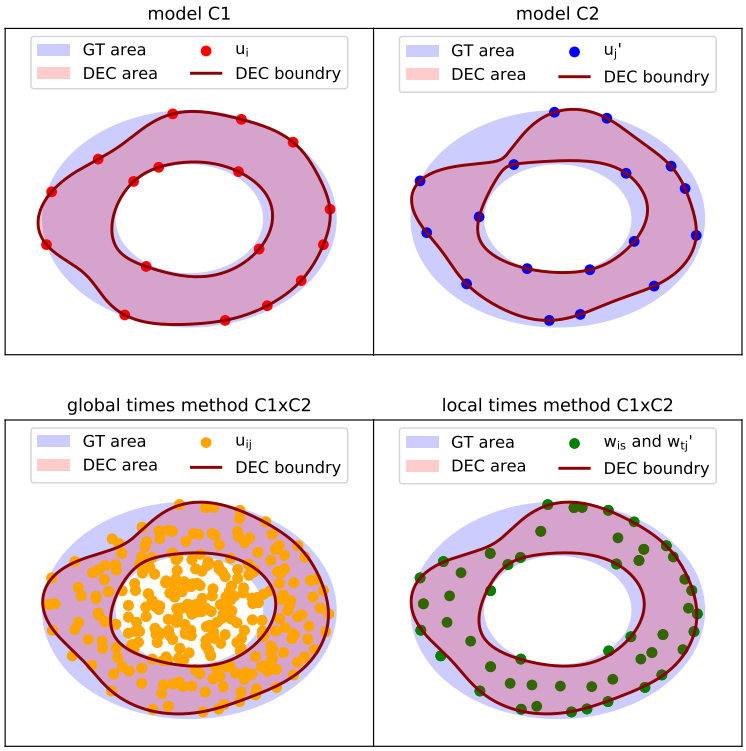

Figure 2.6. Distribution of support vectors of different classifiers in non-convex ground truth space

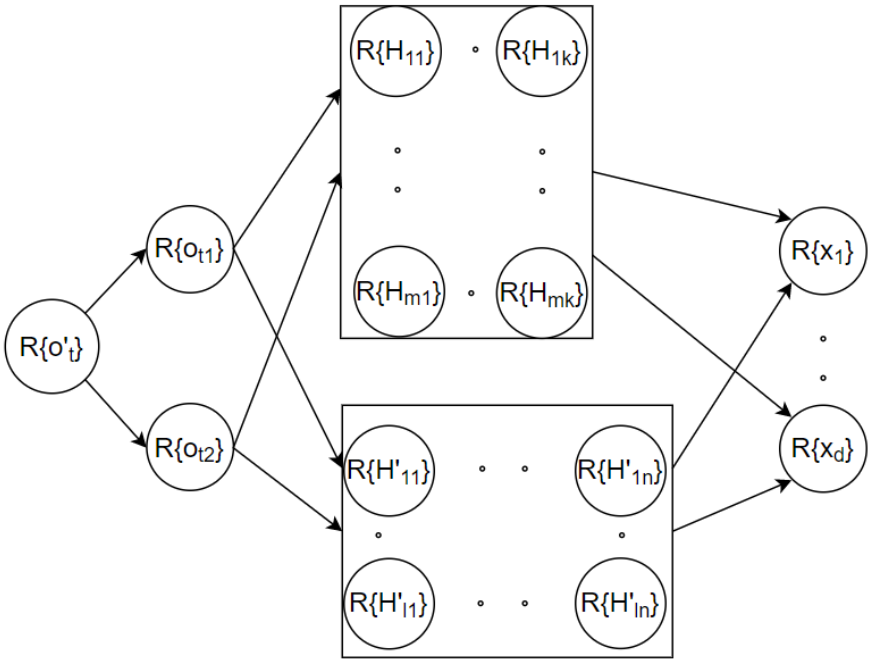

Figure 2.7. The layer-wise relevance propagation for network 4

Compared with "global Times" method, the "local Times" method has the following properties:

1) When $k \neq n, l \neq m$, which means parts of the β_{ij} are ignored (i.e. set as zero). On the one hand we would lose a bit of information of equation (2.10), which may result in the slight decrease of accuracy of the models; on the other hand it would lead to a great improvement of program running time, as $m * n$ neurons grows much faster than $m * k + n * l$ neurons (k, l are constants);

2) Particularly, when $k = n, l = m$, which means none of the β_{ij} in equation (2.10) is ignored (i.e. set as 0). Then $o'_t = o_t$, the "local Times" method is equivalent to the "global Times" method.

The relevance score of the input layer is:

$$\begin{aligned} R_{\{x\}} = & \frac{1}{2}(\frac{\partial D}{\partial x})^2 \cdot [min(o_{t1}, \frac{D}{2} \odot \frac{\partial H}{\partial D}) \odot \frac{\partial o_{t1}}{\partial h}] \\ & + \frac{1}{2}(\frac{\partial D'}{\partial x})^2 \cdot [min(o_{t2}, \frac{D'}{2} \odot \frac{\partial H'}{\partial D'}) \odot \frac{\partial o_{t2}}{\partial H'}] \end{aligned} \tag{2.15}$$

The "local Times" method is more proper for practical applications. The parameters k, l can be used to fine-tune the models. When we set the parameters k, l to low values, it can reduce running time at a slight cost of accuracy; When we set the parameters to their upper bound $k = n, l = m$, the model is identical to "global Times" method.

Chapter 3

Experiments and Conclusion

In this section we compare the performance of "bagging" and "local Times" models. Let C_1 and C_2 be two basic OC-SVM classifiers; $C_x + C_y$, $x, y \in \{1, 2\}$ denotes the model fused in "bagging" method; $C_x \times C_y$, $x, y \in \{1, 2\}$ denotes the model fused in "local Times" method. Thus, based on C_1 and C_2 we generate 6 different classifiers, i.e. $C_1 + C_1$, $C_1 \times C_1$, $C_2 + C_2$, $C_2 \times C_2$, $C_1 + C_2$ and $C_1 \times C_2$.

According to equation (2.3) we know that, for any input data x the relevance score $R_{\{x\}}$ in classifier $C_x + C_x$, $x \in \{1, 2\}$ is always exactly same as that in classifier C_x, which means $C_x + C_x$ is essentially equivalent to C_x. Thus, these six classifiers actually are C_1, $C_1 \times C_1$, C_2, $C_2 \times C_2$, $C_1 + C_2$ and $C_1 \times C_2$.

These models are applied on three different datasets: MNIST, MNIST-C and MVTec. The accuracy of classification and the heatmap of the relevance of input layer are used to evaluate the models.

3.1. Experiment on MNIST

in this section we test the models on MNIST dataset, our goal is to build models those can recognize digits 2, i.e. digits 2 would be classified as inliers while the rest digits would be classified as outliers. Hence, we choose all the digits 2 in MNIST as our train data.

The basic OC-SVM models C_1 and C_2 are built in a special way. Assume there is train set $\{x_i\}, i = 1, ..., m$, the first step is to set the proper value of γ for Gaussian kernel $k(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$. The constraint is that, γ makes the neighbors of data points keep the most information of the entire training set.

Let $x_{i1}, ..., x_{ik}$ be the k-nearest neighbors of data point x_i (in this thesis k is chosen as 10 percents of the number of whole train samples, i.e. $k = 0.1 * m$), then the constraint can be expressed as the formula below:

$$\frac{\sum_{i=1}^{m} \sum_{s=1}^{k} k(x_i, x_{is})}{\sum_{i \neq j} k(x_i, x_j)} \approx 0.9 \tag{3.1}$$

After finding the proper γ value, we train OC-SVM models on two subsets of all

digits 2 separately. As a result of that, we get the two basic OC-SVM classifiers C_1 and C_2 with formula (2.1). Then $C_1 + C_2$, $C_1 \times C_1$, $C_2 \times C_2$ and $C_1 \times C_2$ can be generated.

We test the models on the MNIST testset, the ROC_AUC score is used to evaluate the accuracy of classifiers. The results in Table (3.1) shows that: 1) when we increase m, n (i.e. the numbers of support vectors in C_1 and C_2), the overall trend of model accuracy is increasing, though C_1 and $C_1 \times C_1$ have better performance with $m, n = 50$ than with $m, n = 60$; 2) C_x and $C_x \times C_x$, $x \in \{1, 2\}$ have same accuracy; 3) The accuracy of $C_1 + C_2$ is the highest and slightly better than $C_1 \times C_2$.

Table 3.1. ROC_AUC scores of different classifiers on MNIST

classifier / m, n	C_1	$C_1 \times C_1$	C_2	$C_2 \times C_2$	$C_1 + C_2$	$C_1 \times C_2$
40	0.846	0.846	0.830	0.830	0.860	0.851
50	0.856	0.856	0.851	0.851	0.876	0.869
60	0.851	0.851	0.857	0.857	0.873	0.870
70	0.870	0.870	0.866	0.866	0.885	0.884
80	0.879	0.879	0.870	0.870	0.891	0.889
90	0.885	0.885	0.880	0.880	0.898	0.898
100	0.889	0.889	0.883	0.883	0.902	0.901

By applying equation (1.14), (2.3), (2.9) we could compute R_1, $R_{1\times1}$, R_2, $R_{2\times2}$, R_{1+2} and $R_{1\times2}$, which are the corresponding relevance scores of the input layers related to C_1, $C_1 \times C_1$, C_2, $C_2 \times C_2$, $C_1 + C_2$ and $C_1 \times C_2$. As these classifiers have different outlierness function, it makes no sense if we compare their relevance scores directly. For example, assume there are two well-trained classifiers C_a, C_b and two constants $a, b > 0$, which satisfies two conditions:

1. $\forall x \in R^d$, $R_a(x) = a * R_b(x) + b$, where R_a and R_b are relevance scores of the input layers related to C_a and C_b;

2. $\exists x_{in}, x_{out} \in R^d$, $s.t.$ $R_a(x_{in}) > R_b(x_{out})$, where x_{in} is an inlier and x_{out} is an outlier.

Then from condition 1 we know that, although R_a is always larger than R_b, they will share same heatmap if we plot them individually, which means the explanation of these two models are identical. However, if we evaluate the explanation with relevance scores directly, the condition 2 would lead to the false conclusion that C_a has a worse explanation.

To solve this problem, we add one more step to normalize the relevance scores, i.e. map them to values in interval $[0, 1]$. After the normalization, the new relevance score r_i' can directly be used to measure, which parts of the data point are abnormal and to what extent will these parts be detected as anomaly.

Assume $r_i, i = 1, .., k$ are the relevance scores of k test data points by applying classifier C, $v_{max,i}$ and $v_{min,i}$ are the maximum and minimum element values of vector r_i, then the normalization step is like below:

$$r_i' = \frac{r_i - min\{v_{min,1}, ..., v_{min,k}\}}{max\{v_{max,1}, ..., v_{max,k}\} - min\{v_{min,1}, ..., v_{min,k}\}} \tag{3.2}$$

With these normalized relevance score, we can generate the adjusted heatmaps for the test data. Figure 3.1 shows the heatmaps of some test digits with $m, n = 100$.

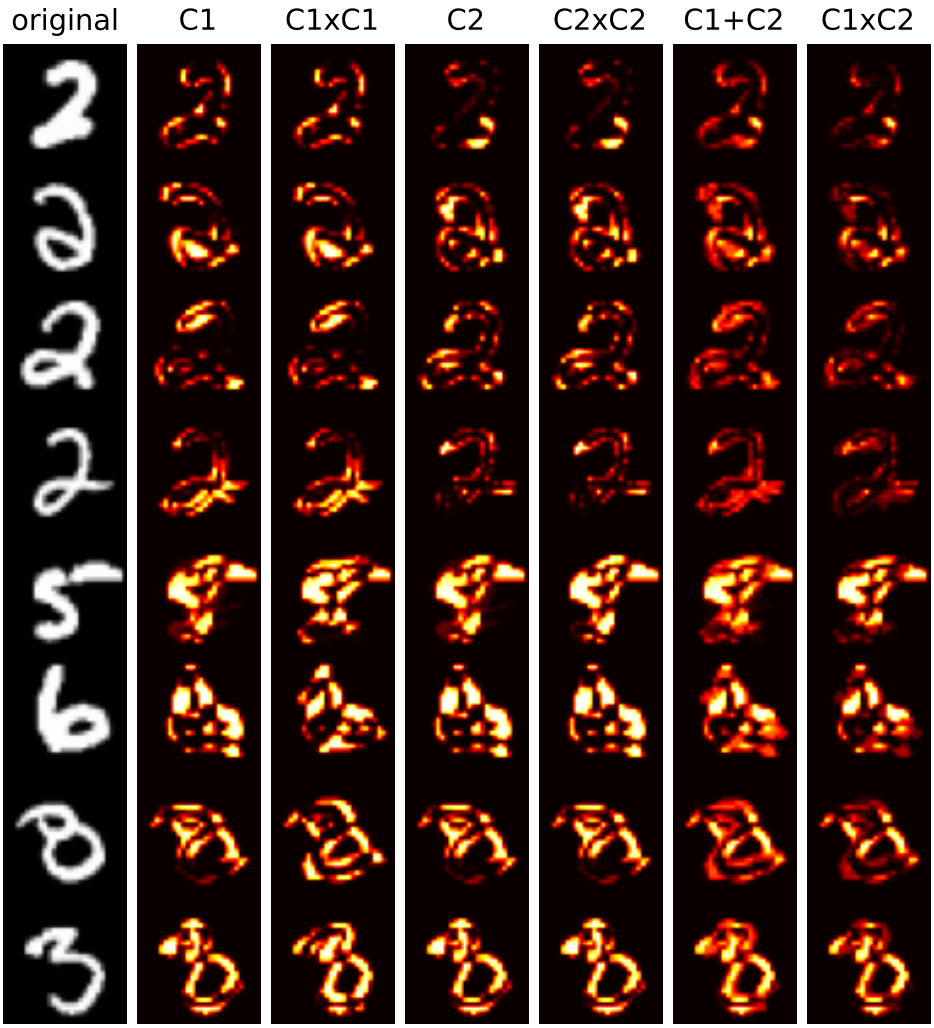

Figure 3.1. Heatmaps of MNIST samples. The first four rows are heatmaps of inliers, the last four rows are heatmaps of outliers.

The heatmaps in Fig 3.1 shows that: 1) In general the digits 2 has less heat than non-2 digits, which means all of these six models can classify the in- and outliers well; 2) C_1 and $C_1 \times C_1$ have similar heatmaps for inlier test samples, while the heatmaps for outlier samples are different; 3) C_2 and $C_2 \times C_2$ have similar heatmaps for all test samples; 4) $C_1 \times C_2$ has lighter heatmaps for inlier samples than $C_1 + C_2$, which means, these digits 2 would more likely be classified as inliers by applying "local Times" method than "bagging" method; 5) $C_1 + C_2$ and $C_1 \times C_2$ have significant different heatmaps for outlier samples.

To evaluate these heatmaps, we apply "Pixel-Flipping" evaluation metric on the heatmaps and their outlierness functions. The result is shown in Fig 3.2.

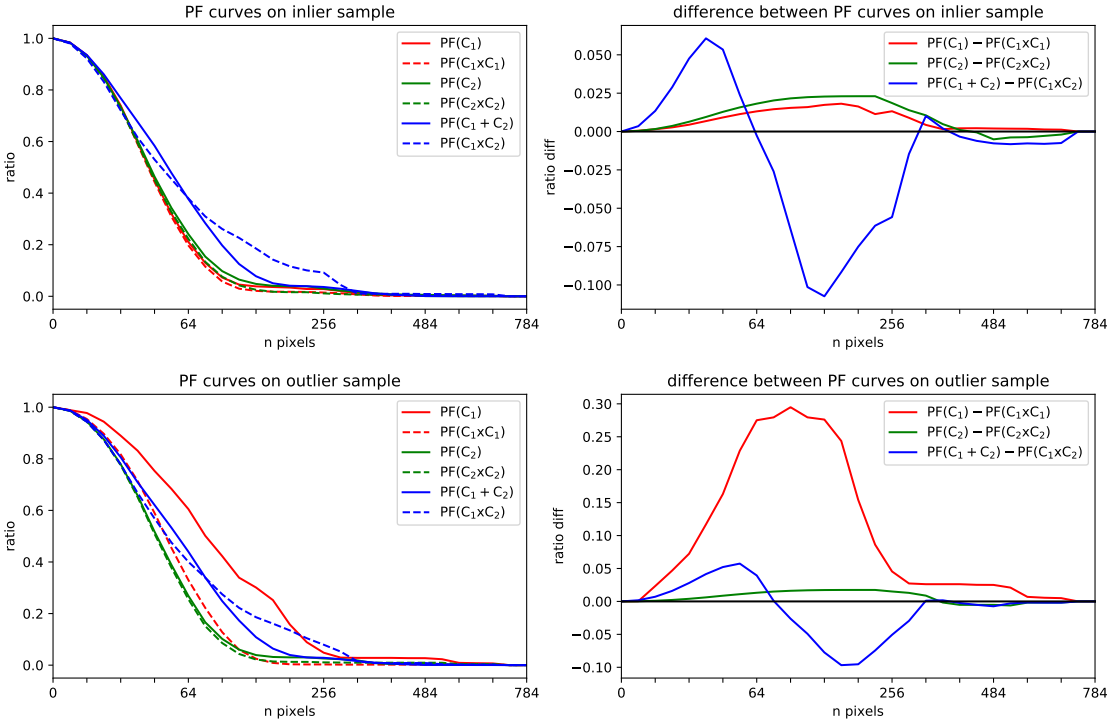

Figure 3.2. PF curves of different combinations of heatmaps and outlierness functions. The "n-pixels" on x-axis means the number of masked pixels; the "ratio" on y-axis means the ratio between the masked and unmasked outlierness score.

From Fig 3.2 we know that: 1) The PF curve of $C_1 \times C_1$ is always under the PF curve of C_1, the difference on inlier sample is small while it is large on outlier sample, which means the explanation of $C_1 \times C_1$ is much better than C_1, especially on the outlier sample; 2) The PF curve of $C_2 \times C_2$ is always under the PF curve of C_2, but the difference is small, which means the explanation of $C_2 \times C_2$ is slightly better than C_2; 3) When n is small, then $PF(C_1 \times C_2) < PF(C_1 + C_2)$. when n is large, then $PF(C_1 \times C_2) > PF(C_1 + C_2)$. Which means that $C_1 \times C_2$ works better on explaining the most relevant pixels, while $C_1 + C_2$ is better at explanation the low relevant pixels.

In practical application, we have more interest on capturing the most abnormal data, i.e. the data with highest relevance score. Thus, we conclude that the "Times" method has an better explanation than "bagging" method on MNIST dataset.

3.2. Experiment on MNIST-C

MNIST-C dataset is a comprehensive suite of 15 corruptions applied to the MNIST data set, for benchmarking out-of-distribution robustness in computer vision[12]. Fig 3.3 shows an example. In this section we want to build OC-SVM models that can distinguish the normal samples in MNIST and the corrupted samples in MNIST-C.

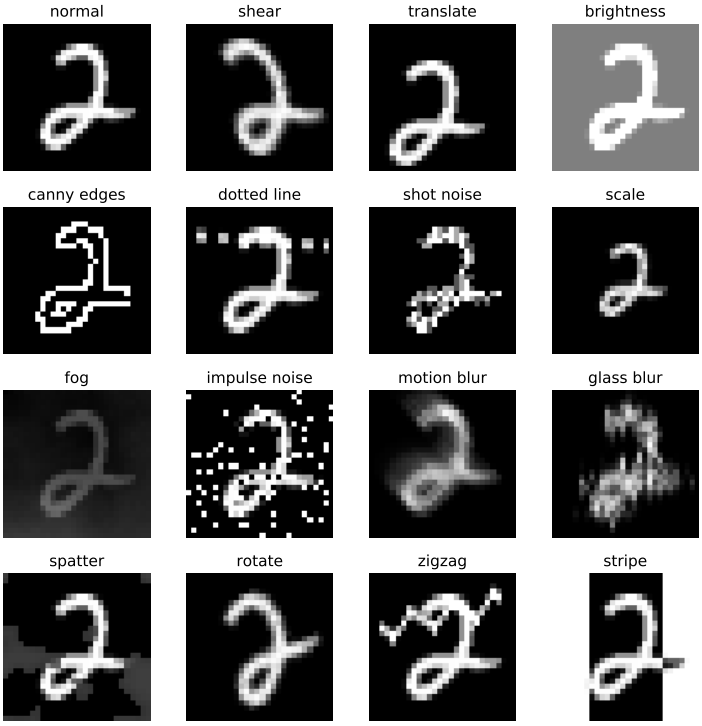

Figure 3.3. Example of MNIST-C. The first digit is a clean data point from MNIST, the rest 15 digits are from MNIST-C, which are the combinations of the clean data point and different corruptions.

We trained the models on the whole MNIST training set, and test these models with the testsets of MNIST and MNIST-C, i.e. the test samples from MNIST are inliers while test samples from MNIST-C are outliers. The ROC_AUC scores of different classifiers are shown in Table (3.2).

Table 3.2. ROC_AUC scores of different classifiers on MNIST-C

label \ classifier	C_1	$C_1 \times C_1$	C_2	$C_2 \times C_2$	$C_1 + C_2$	$C_1 \times C_2$
shear	0.533	0.533	0.597	0.597	0.581	0.582
translate	0.892	0.892	0.903	0.903	0.918	0.915
brightness	1.000	1.000	1.000	1.000	1.000	1.000
canny edges	0.930	0.930	0.958	0.955	0.960	0.960
dotted line	0.708	0.708	0.713	0.713	0.750	0.743
shot noise	0.592	0.592	0.613	0.613	0.622	0.617
scale	0.261	0.261	0.347	0.347	0.260	0.263
fog	0.883	0.883	0.865	0.865	0.877	0.874
impulse noise	0.996	0.996	0.998	0.998	0.999	0.999
motion blur	0.320	0.320	0.341	0.341	0.305	0.310
glass blur	0.321	0.321	0.318	0.319	0.291	0.294
spatter	0.547	0.547	0.553	0.553	0.562	0.560
rotate	0.491	0.491	0.507	0.507	0.500	0.503
zigzag	0.886	0.886	0.868	0.868	0.893	0.893
stripe	0.880	0.880	0.880	0.880	0.915	0.910

The blue numbers in Table (3.2) are the ROC_AUC scores those are less than 0.600, the red numbers are the highest ROC_AUC scores (s.t. larger than 0.600) in each row. The results show that the OC-SVM models trained on MNIST cannot recognize all the kinds of noise in MNIST-C: 1) For samples in MNIST-C with labels "scale", "motion blur" and "spatter", the models more likely to mis-classify them as inliers; 2) For samples with labels "shear", "shot noise" , "spatter" and "rotate" the performance of the models are close to or slightly better than random guess; 3) For samples with labels "translate", "brightness", "canny edges", "dotted line", "impulse noise", "zigzag" and "stripe", the models can detect the corruption efficiently; 4) The highest ROC_AUC scores is always obtained by $C_1 + C_2$.

Next, we use a special metric to judge which models have better explanation for the outliers. Unlike the experiment in section 3.1, the outliers in this experiment are the corrupted samples, which are the combinations of normal samples in MNIST and special corruptions. Hence, we could extract the pixels which are related to the corruptions and generate the corresponding ground truth images for the outliers. Assume $x \in R^{784}$ is a normalized test sample in MNIST, $x' \in R^{784}$ is the sample in MNIST-C, which consists of x and a special corruption. We use $gt(x')$ to denote the vector of ground truth image. The form of $gt(x')$ is obtained like below:

$$gt(x') = (x' - x) \odot (x' - x) \tag{3.3}$$

After extracting the ground truth images, we build the heatmaps for samples in MNIST-C. The results in Fig 3.4 shows that, C_x and $C_x \times C_x$ ($x \in \{1, 2\}$) have similar explanations for samples in MNIST-C, while there is significant difference between the heatmaps generated with $C_1 + C_2$ and $C_1 \times C_2$.

Then we use the cosine similarity(CS) between the heatmap $R_{\{x'\}}$ and the ground truth of corruptions $gt(x')$ to evaluate the explanation for x'. The higher CS value means the better explanation. We only evaluate the heatmaps of the samples, in which the corruptions can be detected (i.e. $ROC\_AUC > 0.600$). The CS scores are shown in Table (3.3). From which we know that, the heatmaps of model $C_1 \times C_2$ always lead to the highest CS scores, i.e. the "Times" method has an better explanation than the bagging method on these subsets of MNIST-C.

Table 3.3. cosine similarity of different classifiers on MNIST-C

label \ classifier	C_1	$C_1 \times C_1$	C_2	$C_2 \times C_2$	$C_1 + C_2$	$C_1 \times C_2$
translate	0.522	0.522	0.638	0.638	0.671	0.726
brightness	0.802	0.802	0.856	0.856	0.902	0.915
canny edges	0.413	0.413	0.368	0.368	0.456	0.496
dotted line	0.341	0.340	0.497	0.510	0.496	0.639
fog	0.877	0.884	0.901	0.903	0.916	0.936
impulse noise	0.779	0.787	0.801	0.801	0.836	0.860
zigzag	0.554	0.554	0.636	0.636	0.660	0.711
stripe	0.915	0.915	0.966	0.971	0.958	0.977

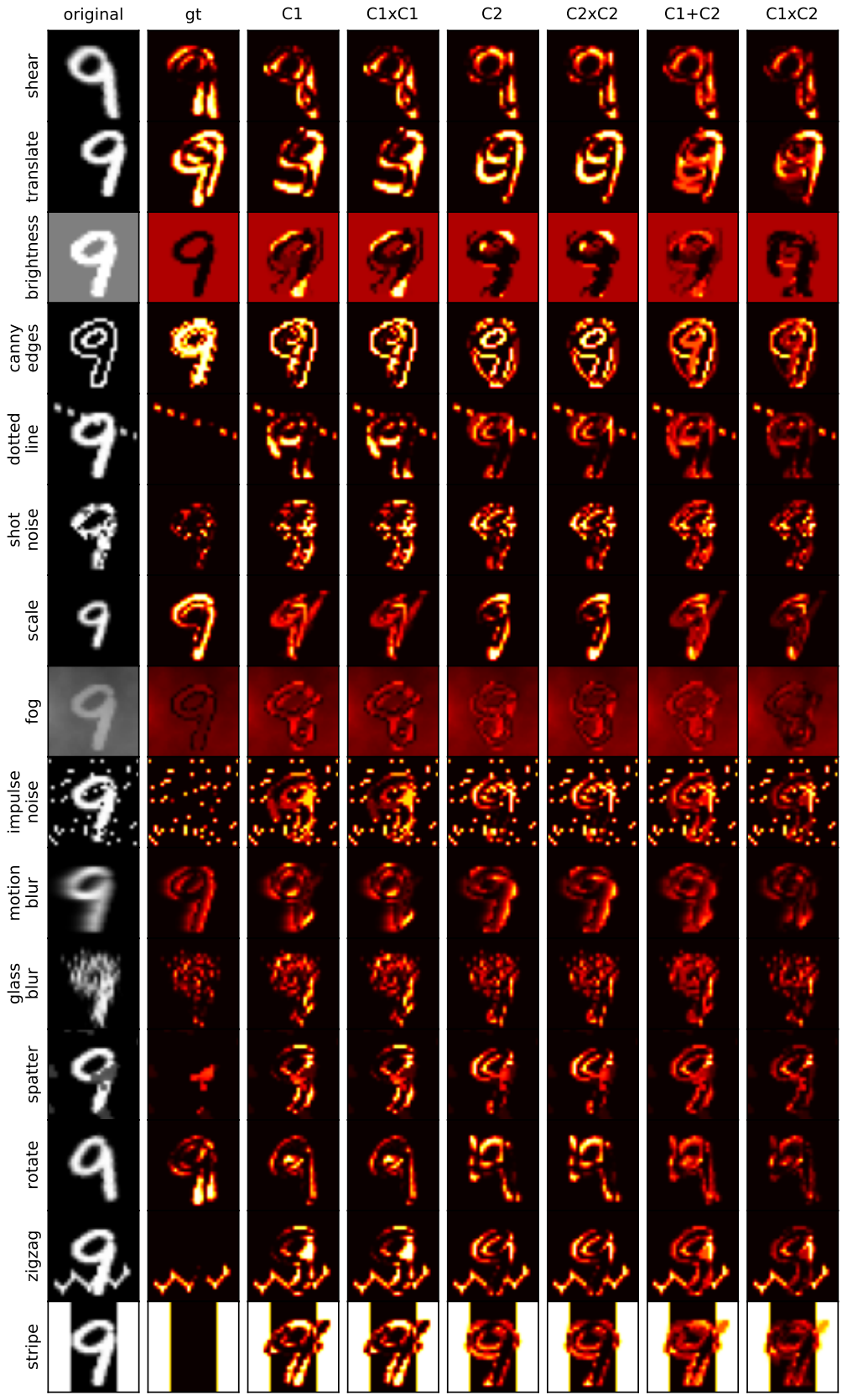

Figure 3.4. Heatmaps of MNIST-C samples

Combined with ROC_AUC scores in Table (3.2), we find that, although $C_1 \times C_2$ has a slightly lower accuracy than $C_1 + C_2$, it gives much better explanation for the anomaly.

3.3. Experiment on MVTec AD

MVTec AD is a dataset for benchmarking anomaly detection methods with a focus on industrial inspection[13]. Unlike the samples in MNIST or MNIST-C data set, the samples in MVTec are high-resolution color images and each of them has about 3 million dimensions ($channels = 3$, $width \approx 1000$, $height \approx 1000$), Fig 3.5 shows some samples in the training sets.

Figure 3.5. The examples of the images in MVTec dataset.

There are only less than 300 samples in each training set. We hereby use a special way to build the basic OC-SVM models C_1 and C_2.

First of all, we lower the images resolution for the training data and the test data. The adjusted images($channels = 3$, $width = 160$, $height = 160$) not only have much less dimensions but also keep enough information of the original images.

Next, we divide the training set T into two subsets T_1 and T_2, s.t. $T_1 \cap T_2 = \varnothing$, $T_1 \cup T_2 = T$, $0 \leq card(T_1) - card(T_2) \leq 1$.

Then, we build the OC-SVM single-model C_i based on set T_i, $i \in \{1, 2\}$. Instead of extracting the support vectors by solving the optimization problem (1.1) that we discussed in section 1.1, all the data points(i.e. the adjusted images) in T_i are treated as support vectors and their corresponding dual coefficients are set as $1/card(T_i)$. This model is also called as kernel density estimation model[26, 27] (KDE) and suit for high-dimensional spaces[28]. The outlierness function of the KDE model could be expressed as below,

$$\begin{aligned} o_i(x) &= -\log\left(\sum_{x_j \in T_i} \frac{1}{card(T_i)} k(x, x_j)\right) \\ &= -\log\left(\sum_{x_j \in T_i} k(x, x_j)\right) + \log\left(card(T_i)\right) \\ &= -\log\left(\sum_{x_j \in T_i} -\gamma||x - x_j||^2\right) + \log\left(card(T_i)\right) \end{aligned} \tag{3.4}$$

where $\log\big(card(S_i)\big)$ is a constant bias and therefore could be removed.

Finally, by applying the "bagging" and "Times" ensemble methods, the ensemble models $C_1 + C_2$, $C_1 \times C_1$,$C_2 \times C_2$, $C_1 \times C_2$ are generated.

The test dataset of MVTec consists of images with or without various kinds of defects, in this section we focus on detecting the anomalies in defected images. Let S_0 denote the set of all the normal images (i.e. inliers) in the test dataset, $S_i (i \in \{1, ..., n\})$ are the n subsets of $MVTec$ test set with different types of defects. Then we test the models on $S_i' = S_i \cup S_0$, again ROC_AUC score is used to evaluate the accuracy of the models on distinguishing the defected images from the normal images.

Table 3.4. ROC_AUC score of different classifiers on MVTec

label \ classifier		C_1	$C_1 \times C_1$	C_2	$C_2 \times C_2$	$C_1 + C_2$	$C_1 \times C_2$
wood	hole	1.000	1.000	1.000	1.000	1.000	1.000
	color	0.987	0.974	0.993	0.993	0.993	0.993
	scratch	0.925	0.917	0.962	0.950	0.942	0.935
	combined	0.909	0.904	0.900	0.900	0.923	0.914
	liquid	0.837	0.832	0.868	0.874	0.853	0.832
leather	poke	0.913	0.913	0.917	0.917	0.917	0.910
	fold	0.781	0.781	0.779	0.779	0.785	0.776
	color	0.673	0.676	0.715	0.719	0.692	0.694
	glue	0.303	0.303	0.322	0.326	0.308	0.312
	cut	0.077	0.081	0.087	0.089	0.082	0.084
tile	crack	0.766	0.736	0.766	0.725	0.770	0.731
	glue strip	0.660	0.650	0.662	0.653	0.660	0.645
	gray stroke	0.672	0.616	0.706	0.623	0.691	0.648
	oil	0.411	0.406	0.414	0.377	0.412	0.392
	rough	0.568	0.576	0.562	0.574	0.453	0.552

Table (3.4) shows the ROC_AUC scores of different models on MVTec. From the table we know: 1) The KDE models have a very good performance on the classification task for test set "wood", especially for the subset with label "hole", the accuracy can even reach to 100% ; 2) For sets "leather" and "tile", the KDE models can only distinguish some specific defects (i.e. the accuracy of which is greater than 0.600); 3) Unlike that the "bagging" method always has the highest accuracy on MNIST and MNIST-C, the "Times" method sometimes leads to the highest ROC_AUC score on MVTec.

Then we generate the heatmaps for the defected samples (i.e. the outliers) in the test sets, the accuracy of which are higher than 0.600. As the pixel-precise ground truth regions for each defected image is provided[13], we could directly use the cosine similarity between the heatmap and the ground truth image to evaluate the explanation of the models.

Fig 3.6 and Fig 3.7 are examples of heatmaps in subset "wood" with labels "hole" and "scratch" respectively. It shows that, although both of them have very high ROC_AUC scores, the models only have a good performance on explaining the regions of "hole", while the explanation for the regions of scratch is much worse, as the models actually

cannot distinguish the texture and the scratch; Fig 3.8 and Fig 3.9 show that, despite the models have better ROC_AUC scores on subsets "leather" with label "poke" than with label "fold", the explanation for the "poke region" is much worse. The reason is that the relevance scores of the texture of leather is very close to the relevance scores of the "color region"; Fig 3.10 and Fig 3.11 are heatmaps of samples in subset "tile with crack" and "tile with gray stroke", the models can detect the most relevant regions of the defects on both of the subsets. Although the heatmaps look noisy, the ground truth regions gain much more heat(i.e relevance score) than the pixels on the background. Therefore, the cosine similarity scores on these subsets are not low.

The average cosine similarity scores of each subset are shown in Table (3.5). The blue numbers are cosine similarity scores those are less than 0.300, which mean the bad explanations. The red numbers are the highest values in each row, which refer to the models with the best explanations.

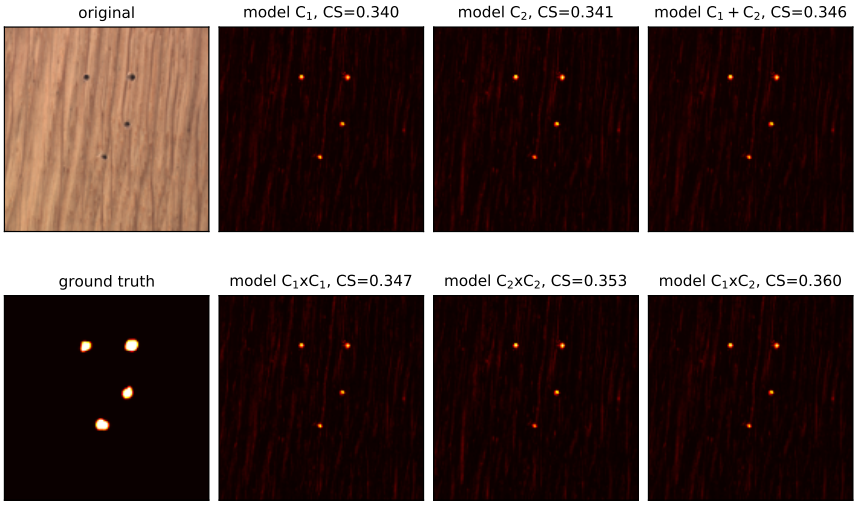

Figure 3.6. Heatmap of wood with hole

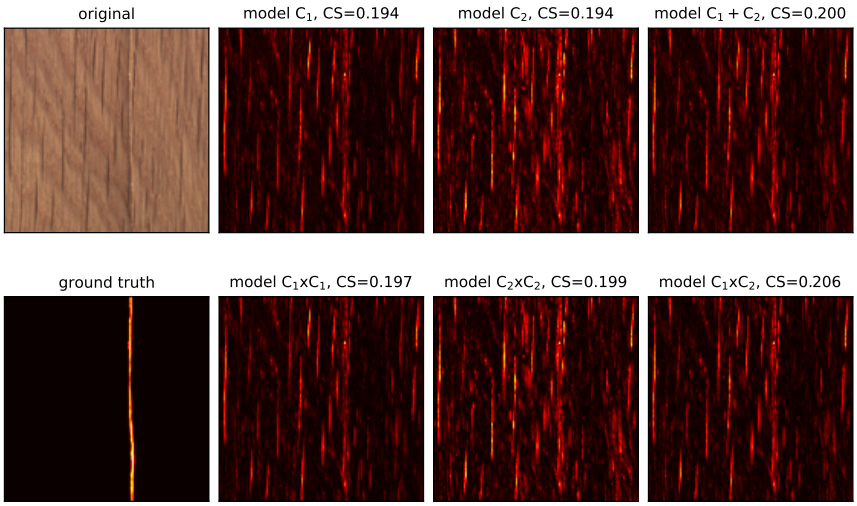

Figure 3.7. Heatmap of wood with scratch

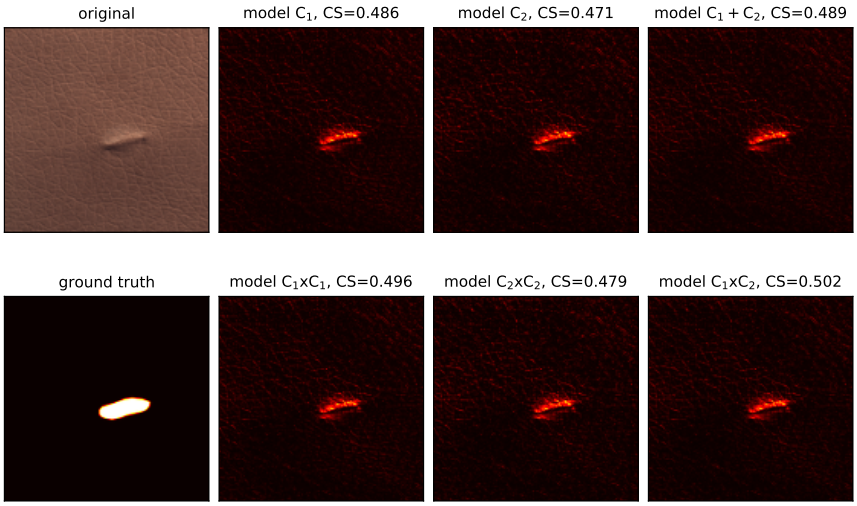

Figure 3.8. Heatmap ofleather with fold

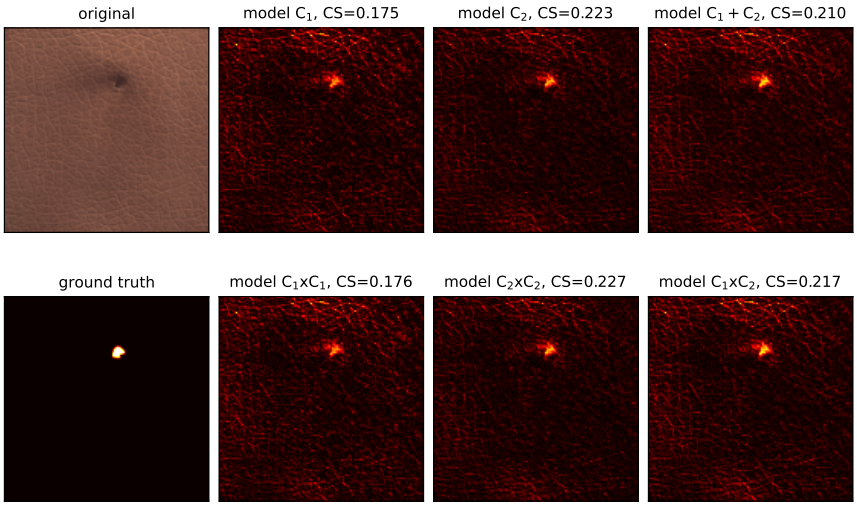

Figure 3.9. Heatmap of leather with color

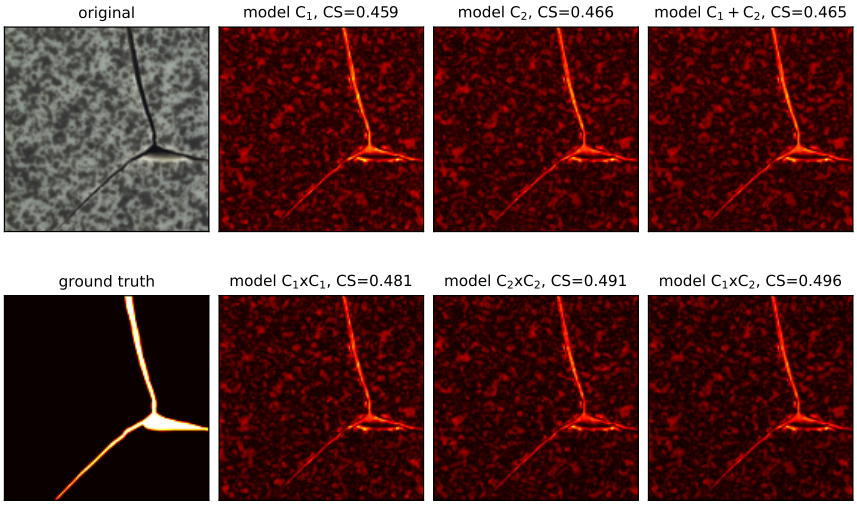

Figure 3.10. Heatmap of tile with crack

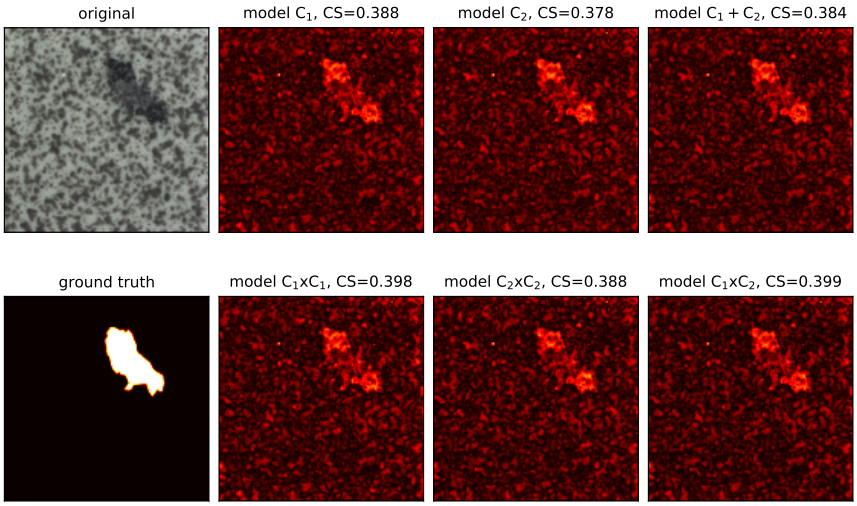

Figure 3.11. Heatmap of tile with gray stroke

Table 3.5. Cosine similarity scores of different classifiers on MVTec

label \ classifier		C_1	$C_1 \times C_1$	C_2	$C_2 \times C_2$	$C_1 + C_2$	$C_1 \times C_2$
wood	hole	0.362	0.367	0.363	0.368	0.365	0.372
	color	0.334	0.332	0.342	0.343	0.342	0.342
	scratch	0.294	0.290	0.295	0.289	0.299	0.296
	combined	0.409	0.409	0.407	0.405	0.413	0.414
	liquid	0.253	0.246	0.263	0.255	0.261	0.254
leather	poke	0.217	0.219	0.216	0.217	0.227	0.231
	fold	0.347	0.349	0.337	0.337	0.354	0.359
	color	0.098	0.098	0.097	0.096	0.102	0.102
tile	crack	0.332	0.351	0.326	0.328	0.327	0.328
	glue strip	0.325	0.322	0.326	0.328	0.327	0.328
	gray stroke	0.338	0.343	0.331	0.335	0.338	0.348

The results in Table (3.5) shows that: 1) The highest average cosine similarity scores is always obtained by the "Times" method. But the superiority of "Times" method over bagging method is less than the experiment results on MNIST-C. The reason is that, the dimensions of samples in MVTec is much higher and the cosine similarity is relative less sensitive for high dimension data; 2) Although the models can achieve the classification task, it may have a terrible explanation for the classification (e.g. the average CS score of set "leather with color" is only around 0.1). This phenomenon is also called as "Clever Hans effect"[29, 30].

Now we define the "Clever Hans score" as the ratio between the ROC_AUC score and the cosine similarity score,

$$Clever\ Hans\ score = \frac{ROC\_AUC\ score}{cosine\ similarity\ score} \tag{3.5}$$

The "Clever Hans score(CH score)" can be used to measure the model's understanding of the classification task. The higher CH score means the worse understanding. In this thesis, we chose $th = 3$ as the thresh hold value to distinguish the models, when $CH < 3$, it means the models have good understanding; otherwise, the models have bad understanding.

We compute the CH score for each subset in Table (3.5), the results are shown in Table (3.6). The blue numbers are the CH scores those are greater than 3. The red numbers are the lowest CH scores in each row, which also indicate the models with the best understanding. The results show that: 1) The lowest CH scores are always obtained by the "Times" method; 2) When a model has a bad explanation (i.e. CS score), it leads to a bad understanding (i.e. CH score).

Table 3.6. Clever Hans scores of different classifiers on MVTec

label	classifier	C_1	$C_1 \times C_1$	C_2	$C_2 \times C_2$	$C_1 + C_2$	$C_1 \times C_2$
wood	hole	2.760	2.725	2.758	2.719	2.737	2.686
	color	2.953	2.934	2.903	2.901	2.905	2.907
	scratch	3.143	3.161	3.263	3.286	3.147	3.161
	combined	2.223	2.221	2.208	2.219	2.236	2.207
	liquid	3.313	3.387	3.306	3.424	3.267	3.270
leather	poke	4.201	4.166	4.253	4.227	4.043	3.933
	fold	2.251	2.240	2.315	2.310	2.220	2.162
	color	6.887	6.923	7.386	7.454	6.774	6.819
tile	crack	2.306	2.099	2.312	2.082	2.308	2.054
	glue strip	2.030	2.015	2.027	1.992	2.017	1.964
	gray stroke	1.991	1.793	2.136	1.862	2.046	1.862

Based on the results of heatmaps, CS scores and CH scores, we make conclusion that, the "local Times" method has a better performance on MVTec AD dataset.

3.4. Conclusion

In this thesis, we build three outlier-ensemble functions to explain the anomalies, i.e. the baseline "bagging" method o_p, the "global Times" method o_t and the "local Times" method $o_{t'}$, where the "global Times" method is a special case of "local Times" method. Considering the running time, the fine-tuned "local Times" method is more suitable in practical application than "global Times" method.

The experiments on MNIST and MNIST-C show that, the "bagging" method and "local Times" method have very close ROC_AUC scores. Although in some situations, the ROC_AUC score in "bagging" method may be slightly higher, the "local Times" method can generate a more reasonable heatmap, i.e. the "local Times" method has a better explanation for the anomalies. The experiment on MVTec AD shows that, compared with "bagging" method, the cosine similarity scores also show that the "local Times" method leads to an better explanation for the anomalies and the "Clever Hans" scores show that it has a better understanding of the prediction results. Thus we conclude that, in practical

application, the "local Times" method is a more suitable outlier-ensemble method than bagging on explaining anomalies.

Acknowledgement

This research was supervised by Jacob Kauffmann, thanks for his valuable discussion and continuous help.

Bibliography

[1] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. ACM Comput. Surv., 41(3):15:1–15:58, 2009.

[2] Marco A. F. Pimentel, David A. Clifton, Lei A. Clifton, and Lionel Tarassenko. A review of novelty detection. Signal Process., 99:215–249, 2014.

[3] Ayaka Masaki, Kent Nagumo, Bikash Lamsal, Kosuke Oiwa, and Akio Nozawa. Anomaly detection in facial skin temperature using variational autoencoder. Artif. Life Robotics, 26(1):122–128, 2021.

[4] Haibo Wu and Shiliang Shi. Real-time anomaly detection in gas sensor streaming data. Int. J. Embed. Syst., 14(1):81–88, 2021.

[5] Bernhard Schölkopf, Robert C. Williamson, Alexander J. Smola, John Shawe-Taylor, and John C. Platt. Support vector method for novelty detection. In Sara A. Solla, Todd K. Leen, and Klaus-Robert Müller, editors, Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999], pages 582–588. The MIT Press, 1999.

[6] Ming Zhang, Boyi Xu, and Dongxia Wang. An anomaly detection model for network intrusions using one-class SVM and scaling strategy. In Song Guo, Xiaofei Liao, Fangming Liu, and Yanmin Zhu, editors, Collaborative Computing: Networking, Applications, and Worksharing - 11th International Conference, CollaborateCom 2015, Wuhan, China, November 10-11, 2015. Proceedings, volume 163 of Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, pages 267–278. Springer, 2015.

[7] Somaieh Amraee, Abbas Vafaei, Kamal Jamshidi, and Peyman Adibi. Abnormal event detection in crowded scenes using one-class SVM. Signal Image Video Process., 12(6):1115–1123, 2018.

[8] Jacob Kauffmann, Klaus-Robert Müller, and Grégoire Montavon. Towards explaining anomalies: A deep taylor decomposition of one-class models. Pattern Recognit., 101:107198, 2020.

[9] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep taylor decomposition. Pattern Recognit., 65:211–222, 2017.

[10] Bach S, Binder A, Montavon G, Klauschen F, Müller K-R, and Samek W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLoS ONE 10(7): e0130140., 2015.

[11] Omer Sagi and Lior Rokach. Ensemble learning: A survey. Wiley Interdiscip. Rev. Data Min. Knowl. Discov., 8(4), 2018.

[12] Norman Mu and Justin Gilmer. MNIST-C: A robustness benchmark for computer vision. CoRR, abs/1906.02337, 2019.

[13] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec AD - A comprehensive real-world dataset for unsupervised anomaly detection. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, pages 9592–9600. Computer Vision Foundation / IEEE, 2019.

[14] Leo Breiman. Bagging predictors. Mach. Learn., 24(2):123–140, 1996.

[15] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: An overview. In Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller, editors, Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, volume 11700 of Lecture Notes in Computer Science, pages 193–209. Springer, 2019.

[16] Maximilian Kohlbrenner, Alexander Bauer, Shinichi Nakajima, Alexander Binder, Wojciech Samek, and Sebastian Lapuschkin. Towards best practice in explaining neural network decisions with lrp. In Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN), 2020.

[17] Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. Layer-wise relevance propagation for neural networks with local renormalization layers. In Alessandro E. P. Villa, Paolo Masulli, and Antonio Javier Pons Rivero, editors, Artificial Neural Networks and Machine Learning - ICANN 2016 - 25th International Conference on Artificial Neural Networks, Barcelona, Spain, September 6-9, 2016, Proceedings, Part II, volume 9887 of Lecture Notes in Computer Science, pages 63–71. Springer, 2016.

[18] Jacob Kauffmann, Malte Esders, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. From clustering to cluster explanations via neural networks. CoRR, abs/1906.07633, 2019.

[19] Wojciech Samek, Alexander Binder, Sebastian Lapuschkin, and Klaus-Robert Müller. Understanding and comparing deep neural networks for age and gender classification. In 2017 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2017, Venice, Italy, October 22-29, 2017, pages 1629–1638. IEEE Computer Society, 2017.

[20] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, volume 70 of Proceedings of Machine Learning Research, pages 3319–3328. PMLR, 2017.

[21] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. IEEE Trans. Neural Networks Learn. Syst., 28(11):2660–2673, 2017.

[22] João B. D. Cabrera, Carlos Gutiérrez, and Raman K. Mehra. Ensemble methods for anomaly detection and distributed intrusion detection in mobile ad-hoc networks. Inf. Fusion, 9(1):96–119, 2008.

[23] Ludmila Kuncheva. A theoretical study on six classifier fusion strategies. IEEE Trans. Pattern Anal. Mach. Intell., 24(2):281–286, 2002.

[24] Aleksandar Lazarevic and Vipin Kumar. Feature bagging for outlier detection. In Robert Grossman, Roberto J. Bayardo, and Kristin P. Bennett, editors, Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and

Data Mining, Chicago, Illinois, USA, August 21-24, 2005, pages 157–166. ACM, 2005.

[25] Hoang Vu Nguyen, Hock Hee Ang, and Vivekanand Gopalkrishnan. Mining outliers with ensemble of heterogeneous detectors on random subspaces. In Database Systems for Advanced Applications, 15th International Conference, DASFAA 2010, Tsukuba, Japan, April 1-4, 2010, Proceedings, Part I, volume 5981 of Lecture Notes in Computer Science, pages 368–383. Springer, 2010.

[26] Emanuel Parzen. On estimation of a probability density function and mode. Ann. Math. Statist., 33(3):1065–1076, 09 1962.

[27] Murray Rosenblatt. Remarks on some nonparametric estimates of a density function. Ann. Math. Statist., 27(3):832–837, 09 1956.

[28] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. Stat. Anal. Data Min., 5(5):363–387, 2012.

[29] Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Unmasking clever hans predictors and assessing what machines really learn. CoRR, abs/1902.10178, 2019.

[30] Jacob Kauffmann, Lukas Ruff, Grégoire Montavon, and Klaus-Robert Müller. The clever hans effect in anomaly detection. CoRR, abs/2006.10609, 2020.

Appendix A

Proofs

A.1. The value of $R_{\{o\}}(h)$ is zero at point $\tilde{h} = h - o$

From the definition we know $R_{\{o\}} := o$, hence,

$$\begin{aligned} R_{\{o\}}(\tilde{h}) := o(\tilde{h}) &= -\log\left(\sum_k \exp(-h_k + o)\right) \\ &= -\log\left(\exp(o)\sum_k \exp(-h_k)\right) \\ &= -\log\left(\exp(o)\cdot\exp(-o)\right) \\ &= 0 \end{aligned} \tag{A.1}$$

A.2. The ϵ in the Taylor decomposition of $R_{\{o\}}$ is zero at point $\tilde{h} = h - o$

With the definition of ϵ we have,

$$\begin{aligned} \epsilon &= R_{\{o\}}(h) - \nabla R_{\{o\}}(\tilde{h})^T(h - \tilde{h}) - R_{\{o\}}(\tilde{h}) \\ &= o - \sum_{i=1}^{m} \frac{\exp(-h_i)}{\sum_{k=1}^{m}\exp(-h_k)} \cdot o - 0 \\ &= o - o = 0 \end{aligned} \tag{A.2}$$

A.3. Asymptotic constancy of w_i, ϵ_i when i dominates the min-pooling

When i dominates the min-pooling, assume τ is a large scalar satisfies constraint (A.3).

$$\forall i' \neq i : h_i < h_{i'} - \tau \tag{A.3}$$

It follows $1 \leq p_i \leq 1/(1 + (m-1)exp(-\tau))$,

$$\lim_{\tau\to\infty} \frac{1}{1 + (m-1)exp(-\tau)} = \frac{1}{1 + (m-1)\cdot 0} = 1 \tag{A.4}$$

It also follows $0 \leq \epsilon_i \leq -\log(1 + (m-1)exp(-\tau))$,

$$\lim_{\tau\to\infty} -\log(1 + (m-1)exp(-\tau)) = -\log(1 + (m-1)\cdot 0) = 0 \tag{A.5}$$

A.4. The relevance scores $R_{\{h\}}$,$R_{\{h'\}}$ are identical in NN1 and in NN2

In this section, we prove that $R_{\{h\}}$ and $R_{\{h'\}}$ of neural network 1 and neural network 2 are the same. First we compute the corresponding relevance score in neural network 1,

$$R_{\{h_k\}} = \frac{\partial o_1}{\partial h_k}\bigg|_{h=\tilde{h}} \cdot (h_k - \tilde{h}_k) = \frac{\exp(-h_k)}{\sum\limits_s \exp(-h_s)} \cdot o_1 \tag{A.6}$$

Define $w_i = \dfrac{\exp(-h_i)}{\sum\limits_s \exp(-h_s)}$,$w'_j = \dfrac{\exp(-h'_j)}{\sum\limits_t \exp(-h'_t)}$ and $\epsilon_i = o_1 - h_i$, $\epsilon'_j = o_2 - h'_j$.

According to equation (1.10), $R_{\{h_k\}}$ can be written as $R_{\{h_k\}} \approx \tilde{R}_{\{h_k\}} = w_k(h_k + \epsilon_k)$, where w_k and ϵ_k both are treated as constant. Similarly, equation (2.5) can be written as $R_{\{H_{ij}\}} \approx \tilde{R}_{\{H_{ij}\}} = w_i w'_j(h_i + h'_j + \epsilon_i + \epsilon'_j)$. Then $\tilde{h}_i = -\epsilon_i$ and $\tilde{h}'_j = -\epsilon'_j$ are the proper root points for $\tilde{R}_{\{H_{ij}\}}$.

Let $R_{\{p\to q\}}$ denote the redistributed relevance from neuron p to neuron q in the lower layer. Because of $H_{ij} = h_i + h'_j$, the relevance score $R_{\{H_{ij}\}}$ can only be redistributed to neuron h_i and h'_j. We compute R_h in neural network 2 as following.

$$\begin{aligned} R_{\{H_{ij}\to h_i\}} &\approx \frac{\partial \tilde{R}_{\{H_{ij}\}}}{\partial h_i}\bigg|_{h=\tilde{h},h'=\tilde{h}'} \cdot (h_i - \tilde{h}_i) \\ &= \frac{\partial w_i w'_j(h_i + h'_j + \epsilon_i + \epsilon'_j)}{\partial h_i}\bigg|_{h=\tilde{h},h'=\tilde{h}'} \cdot (h_i + \epsilon_i) \\ &= w_i w'_j(h_i + \epsilon_i) \end{aligned} \tag{A.7}$$

$$R_{\{h_i\}} = \sum_j R_{\{H_{ij}\to h_i\}} \approx \sum_j w_i w'_j(h_i + \epsilon_i) = w_i(h_i + \epsilon_i) \tag{A.8}$$

It shows that $R_{\{h\}}$ in neural network 1 and neural network 2 would be replaced by the same function $\tilde{R}_{\{h_k\}}$ when applying OC-DTD, i.e. $R_{\{h\}}$ has same value in neural network 1 and neural network 2. Similarly, $R_{\{h'_k\}}$ in neural network 1 and neural network 2 are identical too.