RAPP: NOVELTY DETECTION WITH RECONSTRUCTION ALONG PROJECTION PATHWAY

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose RAPP, a new methodology for novelty detection by utilizing hidden space activation values obtained from a deep autoencoder. Precisely, RAPP compares input and its autoencoder reconstruction not only in the input space but also in the hidden spaces. We show that if we feed a reconstructed input to the same autoencoder again, its activated values in a hidden space are equivalent to the corresponding reconstruction in that hidden space given the original input. We devise two metrics aggregating those hidden activated values to quantify the novelty of the input. Through extensive experiments using diverse datasets, we validate that RAPP improves novelty detection performances of autoencoder-based approaches. Besides, we show that RAPP outperforms recent novelty detection methods evaluated on popular benchmarks.

1 INTRODUCTION

How can we characterize novelty when only normality information is given? Novelty detection is the mechanism to decide whether a data sample is an outlier with respect to the training data. This mechanism is especially useful in situations where a proportion of detection targets is inherently small. Examples are fraudulent transaction detection (Pawar et al., 2014; Porwal & Mukund, 2018), intrusion detection (Lee, 2017; Aoudi et al., 2018), video surveillance (Ravanbakhsh et al., 2017; Xu et al., 2015b), medical diagnosis (Schlegl et al., 2017; Baur et al., 2018) and equipment failure detection (Kuzin & Borovicka, 2016; Zhao et al., 2017; Beghi et al., 2014). Recently, deep autoencoders and their variants have shown outstanding performances in finding compact representations from complex data, and the reconstruction error has been chosen as a popular metric for detecting novelty (An & Cho, 2015; Vasilev et al., 2018). However, this approach has a limitation of measuring reconstruction quality only in an input space, which does not fully utilize hierarchical representations in hidden spaces identified by the deep autoencoder.

In this paper, we propose RAPP, a new method of detecting novelty samples exploiting hidden activation values in addition to the input values and their autoencoder reconstruction values. While ordinary reconstruction-based methods carry out novelty detection by comparing differences between input data before the input layer and reconstructed data at the output layer, RAPP extends these comparisons to hidden spaces. We first collect a set of hidden activation values by feeding the original input to the autoencoder. Subsequently, we feed the autoencoder reconstructed input to the autoencoder to calculate another set of activation values in the hidden layers. This procedure does not need additional training of the autoencoder. In turn, we quantify the novelty of the input by aggregating these two sets of hidden activation values. To this end, we devise two metrics. The first metric measures the total amount of reconstruction errors in input and hidden spaces. The second metric normalizes the reconstruction errors before summing up. Note that RAPP falls back to the ordinary reconstruction-based method if we only aggregate input values before the input layer and the reconstructed values at the output layer.

Also, we explain the motivations that facilitated the development of RAPP. We show that activation values in a hidden space obtained by feeding a reconstructed input to the autoencoder are equivalent to the corresponding reconstruction in that hidden space for the original input. We refer the latter quantity as a hidden reconstruction of the input. Note that this is a natural extension of the reconstruction to the hidden space. Unfortunately, we cannot directly compute the hidden reconstruction as in the computation of the ordinary reconstruction because the autoencoder does not impose any
correspondence between encoding-decoding pairs of hidden layers during the training. Nevertheless, we show that it can be computed by feeding a reconstructed input to the autoencoder again. Consequently, RAPP incorporates hidden reconstruction errors as well as the ordinary reconstruction error in detecting novelty.

With extensive experiments, we demonstrate using diverse datasets that our method effectively improves autoencoder-based novelty detection methods. In addition, we show by evaluating on popular benchmark datasets that RAPP outperforms competing methods recently developed.

Our contributions are summarized as follows.

- We propose a new novelty detection method by utilizing hidden activation values of an input and its autoencoder reconstruction, and provide aggregation functions for them to quantify novelty of the input.
- We provide motivation that RAPP extends the reconstruction concept in the input space into the hidden spaces. Precisely, we show that hidden activation values of a reconstructed input are equivalent to the corresponding hidden reconstruction of the original input.
- We demonstrate that RAPP improves autoencoder-based novelty detection methods in diverse datasets. Moreover, we validate that RAPP outperforms recent novelty detection methods on popular benchmark datasets.

## 2 Related Work

Various novelty detection methods with deep neural networks rely on the reconstruction error (Sakurada & Yairi 2014; Hoffmann 2007; An & Cho 2015), because discriminative learning schemes are not suitable for highly class-imbalanced data which is common in practice. Unsupervised and semi-supervised learnings handle such imbalance by focusing on the characterization of normality and detecting samples out of the normality.

Variational Autoencoders (VAE) (Kingma & Welling 2014) was reported to outperform vanilla autoencoders for novelty detection based on reconstruction error (An & Cho 2015). To carry out the novelty detection outlined in this approach, an autoencoder needs to be trained only with normal data. The autoencoder encodes the training data, which comprises of only normal data in this case, into a lower-dimensional space and decodes them to the input space. To test novelty, an input value is fed to the autoencoder to produce a reconstructed value and calculate the distance between the input and reconstructed values. This distance is the reconstruction error. A higher reconstruction error means that the input value cannot be encoded onto the lower-dimensional space that represents normal data. Therefore, the input value can be marked as a novelty if its reconstruction error exceeds a certain threshold.

Instead of autoencoders, Generative Adversarial Networks (GAN) have been also suggested to model a distribution of normal data (Sabokrou et al. 2018; Schlegl et al. 2017). Despite the same purpose of discovering a simpler, lower-dimensional representation, the training criterion for GAN is focusing on the quality of data generation rather than the reconstruction quality of training data. Recently, several pieces of research have combined autoencoders and adversarial learning to meet both criteria in dimension reduction and data generation (Haloui et al. 2018; Pidhorskyi et al. 2018; Zenati et al. 2018). One limitation of these methods based on the ordinary reconstruction error is that they do not exploit all the information available along the projection pathway of deep autoencoders. We will explain how to leverage this information for novelty detection in the next section.

From the viewpoint of the diversity and ratio of the normal data in novelty detection, there are two cases available. The first case is when a small fraction of classes are normal. This case has been studied in a one-class classification context, and usually evaluated by organizing training data into a collection of samples belonging to a small number of normal classes (Ruff et al. 2018; Perera & Patel 2018; Sabokrou et al. 2018; Golan & El-Yaniv 2018). The second case is when a majority of classes are assigned as normal (An & Cho 2015; Schlegl et al. 2017; Haloui et al. 2018; Zenati et al. 2018). In this case, normal data is more diverse, and the training data is consist of samples of a relatively large number of normal classes: e.g., nine digits of MNIST. One setup does not dominate the other, but depending on applications, either can be more suitable than the other. Different methods may perform differently in both cases. In this paper, we evaluate RAPP and other competing methods with experiments in both setups.
3 PROPOSED METHOD: RAPP

In this section, we describe the proposed novelty detection method RAPP based on an autoencoder. The main idea is to compare hidden activations of an input and its hidden reconstructions along the projection pathway of the autoencoder. To be precise, we project the input and its autoencoder reconstruction onto the hidden spaces to obtain pairs of activation values, and aggregate them to quantify the novelty of the input. For the aggregation, we present two metrics to measure the total amount of difference within each pair.

3.1 RECONSTRUCTION BASED Novelty Detection

An autoencoder $A$ is a neural network consisting of an encoder $g$ and a decoder $f$, responsible for dimension reduction and its inverse mapping to the original input space, respectively: i.e. $A = f \circ g$. For the purpose, training the autoencoder aims to minimize difference between its input $x$ and output $A(x)$. The space that the encoder $g$ constitutes is called the latent space, and provides more concise representation for data than the input space.

Due to this representation learning property, the autoencoder has been widely used for novelty detection. Specifically, training an autoencoder on normal data samples, novelty of a test sample $x$ is measured by the following reconstruction error $\epsilon$:

$$
\epsilon = \|x - A(x)\|_2.
$$

The test sample $x$ is more likely to be novel as the error $\epsilon(x)$ becomes larger, because it means that $x$ is farther from the manifold that the autoencoder describes.

Although this approach has shown a promising result in novelty detection, the reconstruction error alone does not fully exploit information provided by a trained autoencoder especially when its architecture is deep. In other words, hierarchical information identified by the deep architecture is being ignored. This is rather unfortunate because hierarchical representation learning is one of the most successfully proven capabilities of deep neural networks.

To fully leverage that capability, below we will describe the way to exploit hidden spaces to capture the difference between normal and novel samples in more details.

3.2 RECONSTRUCTION ERROR IN HIDDEN SPACES

Let $A = f \circ g$ be a trained autoencoder where $g$ and $f$ are an encoder and a decoder, and $\ell$ be the number of hidden layers of $g$. Namely, $g = g_\ell \circ \cdots \circ g_1$. We define partial computation of $g$ as follows:

$$
g_i = g_i \circ \cdots \circ g_1,
$$

for $1 \leq i \leq \ell$.

Let $x$ be an input vector, and $\hat{x}$ be its reconstruction by $A$: i.e., $\hat{x} = A(x)$. In addition to comparing $x$ and $\hat{x}$ in the input space, as the ordinary approach does, we examine them in hidden spaces along a projection pathway of $A$. More precisely, feeding $x$ and $\hat{x}$ into $A$, we obtain pairs $(h_i, \hat{h}_i)$ of their hidden representations where

$$
h_i(x) = g_i(x),
$$

$$
\hat{h}_i(x) = g_i(\hat{x}) = g_i(A(x)).
$$

Figure 1 illustrates the procedure of computing $h_i$ and $\hat{h}_i$. As a result, novelty of the sample $x$ is quantified by aggregating $H(x) = \{(h_i(x), \hat{h}_i(x)) : 1 \leq i \leq \ell\}$.

The overall procedure of RAPP is summarized in Algorithm 1. To clearly state the required variables to construct $H$, we write the algorithm with the for loop in Lines 3–5, but in practice, all of them can be computed by feed-forwarding one time each of $x$ and $\hat{x}$ to $g$. Note that RAPP is indeed a generalization of the ordinary reconstruction method with defining $g_0$ as the identity function and $s_{ord}$ as follows.

$$
s_{ord}(H(x)) = \|h_0(x) - \hat{h}_0(x)\|_2^2,
$$

where $h_0(x) = g_0(x) = x$ and $\hat{h}_0(x) = g_0(\hat{x}) = \hat{x}$.
Algorithm 1: RAPP to compute a novelty score.

**Input**: Sample $x$, trained autoencoder $A = f \circ g$, the number of layers $\ell$, and aggregation $s$.

**Output**: Novelty score $S$.

1. $\hat{x} = A(x)$.
2. $H = \emptyset$.
3. For each $i$ in 1 to $\ell$ do
   4. $H = H \cup \{(g_i(x), g_i(\hat{x}))\}$.
4. End
5. $S = s(H)$.

In this paper, we provide two metrics $s_{SAP}$ and $s_{NAP}$ which more extensively utilize $H$ than $s_{ord}$. Those are especially suited for the case of zero-knowledge to interpret identified hidden spaces, which commonly happens when modeling with deep neural networks. Note that, however, more elaborate metrics can be designed if we have knowledge on or can characterize the spaces.

3.2.1 **SIMPLE AGGREGATION ALONG PATHWAY (SAP)**

This is the most straightforward metric that one can define on $H$. For a data sample $x$, SAP is defined by summing the square of Euclidean distances for all pairs in $H$:

$$s_{SAP}(x) = \sum_{i=0}^{\ell} \| h_i(x) - \hat{h}_i(x) \|_2^2 = \| h(x) - \hat{h}(x) \|_2^2,$$

where $h(x)$ and $\hat{h}(x)$ are the concatenations of $[h_0(x), \cdots, h_\ell(x)]$ and $[\hat{h}_0(x); \cdots; \hat{h}_\ell(x)]$, respectively.

3.2.2 **NORMALIZED AGGREGATION ALONG PATHWAY (NAP)**

Although SAP is intuitive, it does not consider properties of hidden spaces; distance distributions of pairs in $H$ may be different depending on the individual hidden spaces. For instance, the magnitude of distances can depend on layers, or there may exist correlated neurons even across layers which are unintentionally emphasized in SAP. To capture clearer patterns, we propose to normalize the distances via two steps: orthogonalization and scaling.

Let $d(x) = h(x) - \hat{h}(x)$; given a training set $X$, let $D$ be a matrix whose $i$-th row corresponds to $d(x_i)$ for $x_i \in X$, and $\bar{D}$ be the column-wise centered matrix of $D$. For the normalization, we compute $D = U \Sigma V^T$, SVD of $D$, to obtain its singular values $\Sigma$ and right singular vectors $V$. For a given data sample $x$, we define $s_{NAP}$ as follows:

$$s_{NAP}(x) = \|d(x)^T V \Sigma^{-1}\|_2^2,$$

where $d(x)$ is expressed as a column vector.
Figure 2: Motivation of RAPP. The quantity that RAPP computes, the hidden activation of the reconstruction input, is equivalent to the hidden reconstruction of the input. Since $\tilde{f} = f$, computing $\hat{h}_2' = \hat{h}_2$ does not require explicitly evaluating $\tilde{f}_i$ but only $g_i$ and $f = \tilde{f}$.

4 MOTIVATION OF RAPP

One natural question in using the ordinary reconstruction method is as follows: why do we investigate only the input space? Or, why do we not use information in hidden spaces? While the reconstruction error in the input space is extensively employed, any similar concept does not exist for hidden spaces. There is no explicit correspondence between hidden layers in $g$ and $f$, and as such two spaces defined by a pair of the corresponding hidden layers in $g$ and $f$ cannot be directly compared. Therefore, we cannot directly compute reconstructions in hidden spaces corresponding to activations in those hidden spaces.

Nevertheless, we will show that there is an indirect way to compute the hidden reconstructions without modifying an ordinary autoencoder nor requiring additional components incorporating it. Precisely, we will show below that $\hat{h}_i$ is indeed equivalent to the hidden reconstruction. The overall mechanism is depicted in Figure 2.

4.1 COMPUTATION OF HIDDEN RECONSTRUCTION

Let $A = f \circ g$ be an autoencoder, and $M_0 = \{A(x) : x \in \mathbb{R}^n\}$ be the low dimensional manifold that $A$ describes (Pidhorskyi et al., 2018); i.e.,

$$\forall x \in M_0, \quad x = A(x).$$

Defining $M_i = \{g_i(x) : x \in M_0\}$, which is the low dimensional image of $M_0$ defined by $g_i$, $g$ and $f$ restricted on $M_0$ and $M_i$, respectively, are inverse functions of each other.

Let us assume that there exists a decoder $\tilde{f}_i = \tilde{f}_1 \circ \cdots \circ \tilde{f}_i$ such that

$$\forall x \in M_i, \quad \tilde{f}(x) = f(x),$$

$$\forall h \in M_i, \quad h = (g_i \circ \tilde{f}_i)(h). \tag{2}$$

The second condition makes $\tilde{f}_{i+1}$ a proper decoder corresponding to $g_{i+1}$, and then the $i$-th hidden reconstruction $\hat{h}_i'(x)$ is defined by

$$\hat{h}_i'(x) = (\tilde{f}_{i+1} \circ g_{i+1})(h_i(x)).$$

Finally, we can conclude that $\hat{h}_i(x)$ is equal to $\hat{h}_i'(x)$ for $x \in M_0$ as follows.

$$\hat{h}_i'(x) = (\tilde{f}_{i+1} \circ g_{i+1})(h_i(x)) = (\tilde{f}_{i+1} \circ g)(x) = (g_i \circ \tilde{f} \circ g)(x) = (g_i \circ f)(x) = h_i(\hat{x}) = \hat{h}_i(x),$$

(by equation 2)

where we do not need $\tilde{f}_i$ for computation, but only $g_i$ and $f$. Note that for $x \in M_0$ already on the manifold, its $i$-th hidden reconstruction $\hat{h}_i'(x)$ becomes equal to its hidden activation $h_i(x) = \hat{h}_i(x)$.
for every $1 \leq i \leq \ell$: i.e. $h_i(x) = \hat{h}_i(x)$ as $x = A(x)$. For $x \notin M_0$, its hidden reconstruction $\hat{h}_i(x)$ will differ from it hidden activation $h_i(x)$.

Now, we show the existence of $\hat{f}$ as follows. Since $x = A(x)$ for $x \in M_0$, $g_i$ and $f_i$ are one-to-one functions from $M_{\ell-1}$ and $M_{\ell}$, respectively. Let us define $\tilde{f}_i = g_i^{-1}$ for $M_i$ and $\hat{f} = \tilde{f}_1 \circ \cdots \circ \tilde{f}_\ell$; then it also holds $\hat{f} = g^{-1}$. This implies $x = (\tilde{f} \circ g)(x)$ for $x \in M_0$, and consequently, $\tilde{f} = f$ on $M_\ell$. This definition of $\tilde{f}_i$ satisfies the two conditions above, and as discussed, we can compute hidden reconstructions of an input, $\hat{h}_i(x) = (\tilde{f}_{\ell+1} \circ g)(x)$, through computing the $i$-th hidden activation of the reconstructed input $\hat{x} = A(x)$, $\hat{h}_i(x)$.

**Implementation of $\hat{f}$ by Neural Networks** Given $g_i$, if the symmetric architecture for $\tilde{f}_i$ is used, we may not be able to learn $\tilde{f}_i = g_i^{-1}$. Neural networks are, however, highly flexible frameworks in which we can deal with models of arbitrary function forms by adjusting network architecture. This property enables us to design a layer capable of representing $\tilde{f}_i$. For instance, even if $\tilde{f}_i$ is too complicated to be represented with a single fully connected layer, we can still approximate $\tilde{f}_i$ by stacking multiple layers. Hence, given $g_i$ and $X$, $\tilde{f}_i$ can be represented by neural networks.

## 5 Evaluation

In this section, we evaluate RAPP in comparison to existing methods. To this end, we tested the methods on several benchmarks and diverse datasets collected from Kaggle and the UCI repository which are suitable for evaluating novelty detection methods.

### 5.1 Datasets and Problem Setups

The datasets from Kaggle and the UCI repository are chosen from problem sets of anomaly detection and multi-class classification, summarized in Table 1. We note that MI-F and MI-V share the same feature matrix, but are considered to be different datasets because their labels normal and abnormal are assigned by different columns: i.e. machine completed and pass visual inspection, respectively. We use these datasets to compare RAPP with standard autoencoder-based methods described in Section 5.2.

To compare RAPP with novelty detection methods in recent literatures, we also use popular benchmark datasets for evaluating deep learning techniques: MNIST [LeCun & Cortes, 2010] and F-MNIST [Xiao et al., 2017]. For theses datasets, we do not take pre-split training and test sets, but instead merge them for post-processing. Further setups are described in Section 5.1.

Novelty detection detects novel patterns by focusing on deviations from model-learned normal patterns. Thus, training sets contain only normal samples and test sets contain both normal and anomaly samples in our evaluation setups. Precisely, if a dataset contains an anomaly label, we assign all samples with that label to the test set for detection. If a dataset does not have any anomaly labels, we consider the following two setups.

---

<table>
<thead>
<tr>
<th>Name</th>
<th># Samples</th>
<th># Features</th>
<th># Class</th>
<th>Domain</th>
<th>Novelty Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>MI-F</td>
<td>25,286</td>
<td>58</td>
<td>2</td>
<td>CNC milling</td>
<td>Machine not completed</td>
</tr>
<tr>
<td>MI-V</td>
<td>23,125</td>
<td>58</td>
<td>2</td>
<td>CNC milling</td>
<td>Workpiece out-of-spec</td>
</tr>
<tr>
<td>EOPT</td>
<td>90,515</td>
<td>20</td>
<td>2</td>
<td>Storage system</td>
<td>System failures</td>
</tr>
<tr>
<td>NASA</td>
<td>4,687</td>
<td>33</td>
<td>2</td>
<td>Astronomy</td>
<td>Hazardous asteroids</td>
</tr>
<tr>
<td>FARM</td>
<td>20,221</td>
<td>6</td>
<td>2</td>
<td>Robotics</td>
<td>Malfunctions</td>
</tr>
<tr>
<td>SILL</td>
<td>1,941</td>
<td>27</td>
<td>7</td>
<td>Steel</td>
<td>Surface defects</td>
</tr>
<tr>
<td>OTTO</td>
<td>61,878</td>
<td>93</td>
<td>9</td>
<td>E-commerce</td>
<td>Types of products</td>
</tr>
<tr>
<td>STL</td>
<td>58,509</td>
<td>48</td>
<td>11</td>
<td>Electric Currents</td>
<td>Defective conditions</td>
</tr>
<tr>
<td>MNIST</td>
<td>70,000</td>
<td>784</td>
<td>10</td>
<td>Hand written digits</td>
<td>Digits</td>
</tr>
<tr>
<td>F-MNIST</td>
<td>70,000</td>
<td>784</td>
<td>10</td>
<td>Fashion articles</td>
<td>Articles</td>
</tr>
</tbody>
</table>

---

Table 1: Description of datasets used in our evaluation.
• Multimodal Normality: A single class is chosen to be the novelty class and the remaining classes are assigned as the normal class. This setup is repeated to produce sub-datasets with all possible novelty assignments. For instance, MNIST results in a set of datasets with 10 different novelty classes.

• Unimodal Normality: In contrast to the multimodal normality setup, we take one class for normality, and the others for novelty. For instance, MNIST results in a set of datasets with 10 different normal classes.

We applied these two setups to STL, OTTO, SNSR, MNIST, and F-MNIST datasets.

5.2 Comparison Method

We compare RAPP and the other methods using Area Under Receiver Operating Characteristic (AUROC). Note that we do not employ thresholding-based metrics such as $F_1$ score because access to abnormal samples is only allowed in testing time. Hence, we focus on the separability of models for novelty with AUROC.

For the datasets in Table 1, we compare the effectiveness of the reconstruction error, SAP and NAP for three models: Autoencoder (AE), Variational Autoencoder (VAE), Adversarial Autoencoder (AAE) (Makhzani et al., 2016). For the benchmark datasets, recent approaches including OCNN (Chalapathy et al., 2018), GPND (Pidhorskyi et al., 2018), DSVDD (Ruff et al., 2018) and GT (Golan & El-Yaniv, 2018) are available. To obtain the performances of the existing approaches, we downloaded their codes and applied against our problem setups.

Given novelty classes, we create the test sets by randomly selecting samples while maintaining novelty ratios to 35% for the multimodal and 50% for the unimodal normality setups, respectively. Note that the expectation value of AUROC is invariant to the novelty ratio.

5.3 Implementation Details

We use symmetric architecture with fully-connected layers for the three base models, AE, VAE, and AAE. Each encoder and decoder has 10 layers with different bottleneck size. For the Kaggle and UCI datasets, we carry out PCA for each dataset first. The minimum number of principal components that explain at least 90% of the variance is selected as the bottleneck size of the autoencoders. We set bottleneck size to 20 for benchmark datasets. Leaky-ReLU (Xu et al., 2015a) activation and batch normalization (Ioffe & Szegedy, 2015) layers are appended to all layers except the last layer.

We train AE, VAE and AAE with Adam optimizer (Kingma & Ba, 2015), and select the model with the lowest validation loss as the best model. For training stability of VAE, 10 Monte Carlo samples were averaged in the reparamterization trick (Kingma & Welling, 2014) to obtain reconstruction from the decoder. In the calculation of SAP and NAP, we excluded reconstructions in the input space for MNIST and F-MNIST.

5.4 Results

Each AUROC score is obtained by averaging AUROC scores from five trials to reduce the random errors in training neural networks.

5.4.1 Comparison with Baselines

Table 2 summarizes the result of our performance evaluation where the best score for each model is in bold. Also, we showed the best score for each dataset with an underline.

Since STL, OTTO, SNSR, MNIST, and F-MNIST do not have anomaly labels, their scores are averaged over all possible anomaly class assignments. For instance, the AUROC value for OTTO in the unimodal normality setup is the average of 9 AUROC values with different anomaly class assignments.

In Table 2, RAPP shows the highest AUROC scores for most of the cases. We also observe that NAP is more effective with AE and VAE compared to AAE. If we examine the performance for each dataset, RAPP achieves the best for 13 cases out of 15 (see the underlines).
Table 2: AUROC of RAPP and the baselines.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>AE</th>
<th>VAE</th>
<th>AAE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Recon</td>
<td>SAP</td>
<td>NAP</td>
</tr>
<tr>
<td>STL</td>
<td>0.596</td>
<td>0.603</td>
<td><strong>0.714</strong></td>
</tr>
<tr>
<td>OTTO</td>
<td>0.620</td>
<td>0.630</td>
<td><strong>0.662</strong></td>
</tr>
<tr>
<td>SNSR</td>
<td>0.601</td>
<td>0.611</td>
<td><strong>0.645</strong></td>
</tr>
<tr>
<td>MNIST</td>
<td>0.825</td>
<td>0.881</td>
<td><strong>0.899</strong></td>
</tr>
<tr>
<td>F-MNIST</td>
<td>0.712</td>
<td>0.725</td>
<td><strong>0.734</strong></td>
</tr>
</tbody>
</table>

Table 3: AUROC on benchmark datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>OCNN</th>
<th>GPND</th>
<th>DSVDD</th>
<th>GT</th>
<th>NAP_{AE}</th>
<th>NAP_{VAE}</th>
<th>NAP_{AAE}</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>0.600</td>
<td>0.501</td>
<td>0.622</td>
<td>0.893</td>
<td>0.899</td>
<td>0.927</td>
<td><strong>0.929</strong></td>
</tr>
<tr>
<td>F-MNIST</td>
<td>0.609</td>
<td>0.691</td>
<td>0.610</td>
<td>0.725</td>
<td>0.734</td>
<td><strong>0.737</strong></td>
<td>0.727</td>
</tr>
</tbody>
</table>

5.4.2 COMPARISON WITH COMPETITORS

Table 3 summarizes the comparison of RAPP to recent novelty detection methods. As in Table 2, AUROC values are calculated by averaging results from 10 cases with different anomaly class assignments for both datasets.

Except for the unimodal F-MNIST setup, NAP outperforms all competing methods regardless of base model choice. Even in the case where NAP scores did not win, the performance of RAPP is comparable to the best one, GT, which relies on image-specific data transformations.

6 CONCLUSION

In this paper, we propose a novelty detection method which utilizes hidden reconstructions along a projection pathway of deep autoencoders. To this end, we extend the concept of reconstruction in the input space to hidden spaces found by an autoencoder and present a tractable way to compute the hidden reconstructions, which requires neither modifying nor retraining the autoencoder. Our experimental results show that the proposed method outperforms other competing methods in terms of AUROC for diverse datasets including popular benchmarks.
REFERENCES


MNIST. http://yann.lecun.com/exdb/mnist/.


Dan Xu, Elisa Ricci, Yan Yan, Jingkuan Song, and Nicu Sebe. Learning deep representations of appearance and motion for anomalous event detection. In BMVC, 2015b.
