Deep Active Learning for Anomaly Detection

Tiago Pimentel
Kunumi
Belo Horizonte, Brazil
tpimentelms@gmail.com

Marianne Monteiro
Kunumi
Campina Grande, Brazil
mariannelinharesm@gmail.com

Adriano Veloso
CS Dept@UFMG
Belo Horizonte, Brazil
adrianov@dcc.ufmg.br

Nivio Ziviani
CS Dept@UFMG & Kunumi
Belo Horizonte, Brazil
nivio@dcc.ufmg.br

***Abstract*—Anomalies are intuitively easy for human experts to understand, but they are hard to define mathematically. Therefore, in order to have performance guarantees in unsupervised anomaly detection, priors need to be assumed on what the anomalies are. By contrast, active learning provides the necessary priors through appropriate expert feedback. Thus, in this work we present an active learning method that can be built upon existing deep learning solutions for unsupervised anomaly detection, so that outliers can be separated from normal data effectively. We introduce a new layer that can be easily attached to any deep learning model designed for unsupervised anomaly detection to transform it into an active method. We report results on both synthetic and real anomaly detection datasets, using multi-layer perceptrons and autoencoder architectures empowered with the proposed active layer, and we discuss their performance on finding clustered and low density anomalies.**

***Index Terms*—Anomaly Detection, Active Learning, Deep Learning**

I. Introduction

Anomaly detection (a.k.a. outlier detection) [1]–[3] aims to discover instances that do not conform to the patterns of majority. The key challenge in anomaly detection applications is that sufficient anomalies and correct labels are often prohibitively expensive to acquire. This problem has been amply studied [4]–[6], with solutions inspired by extreme value theory [7], robust statistics [8] and graph theory [9].

Given that label acquisition is expensive and time consuming, anomaly detection is often applied on unlabeled data which is known as unsupervised anomaly detection. It is a specially hard task, since there is no information on what anomalous instances are. Still, there is a rising trend of adopting unsupervised anomaly detection, with most works using models with implicit priors or heuristics to discover anomalies and providing an anomaly score $s(x)$ for each instance in a dataset. Active anomaly detection is a powerful alternative approach to this problem, which has presented good results in recent works [10]–[14]. The basic idea is that experts can give feedback, thus indicating a few anomaly examples. The subset of anomalies provides valuable input in order to learn better representations of what is normal and anomalous.

Since unsupervised anomaly detection requires priors to be assumed on the anomaly distribution, we approach the anomaly detection problem with an active learning method which we call UAI − Unsupervised to Active Inference. UAI is a layer that can be applied on top of any unsupervised anomaly detection deep learning model to transform it into an active model. The UAI layer is a classifier trained on usually few already labeled instances using the strongest assets of deep unsupervised anomaly detection models: the learned latent representations coupled with an anomaly score.

Experimental results show that the UAI layer applied on top of simple deep unsupervised anomaly detection architectures outperforms state-of-the-art anomaly detection methods on several synthetic and real datasets. We compare the anomaly detection performance of our models against unsupervised, semi-supervised and active competitors under similar budgets. This is done without needing hyper-parameter tuning, as in practical applications, we would not have (initially) any labeled data on which we could tune hyper-parameters, or have too few labeled instances to do it reliably. We also visualize our models' learned latent representations and compare them to the ones from unsupervised models.

II. Problem Definition

In [15], the authors define an outlier as one that appears to deviate markedly from other members of the sample in which it occurs. In [16], the authors state that an outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism. Another definition appears in [2]: "normal" data points occur in high probability regions of a stochastic model, while anomalies occur in the low probability ones.

Following these definitions, we assume there is a probability density function from which "normal" data points are generated as $p_{normal}(x) = p(x|y=0)$, where x is a data point and y is a label saying if the point is anomalous or not. There is also a different probability density function from which anomalous points are sampled as $p_{anom}(x) = p(x|y=1)$. A full dataset is composed of points sampled from the probability distribution that follows:

$$\begin{aligned} p_{full}(x,y) &= p(y)\,p(x|y) \\ p_{full}(x) &= (1-\lambda)p_{normal}(x) + \lambda p_{anom}(x) \end{aligned}$$

where λ is an usually small constant representing the probability of a random data point being anomalous. In [2], the authors divide anomaly detection learning systems in three different types:

We thank the partial support given by the Project: Models, Algorithms and Systems for the Web (grant FAPEMIG / PRONEX / MASWeb APQ-01400-14), and authors' individual grants and scholarships from CNPq, Fapemig and Kunumi. TP and MM did this work while at Kunumi, TP is now at University of Cambridge and MM is now at DeepMind.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

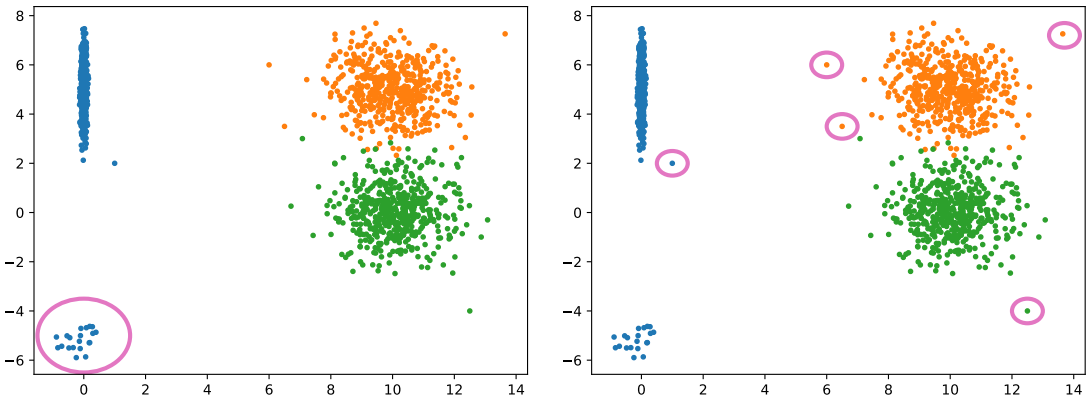


Fig. 1: Illustrative example of undecidable anomaly distribution. (Left) Anomalies are clustered. (Right) Low density anomalies.

- Supervised: A training and a test set are available with curated labels for normal and anomalous points. This case is similar to an unbalanced supervised setting:

$$\mathcal{D}_{train/test} = (X, Y)_{train/test} \sim p_{full}(x, y)$$

- Semi-Supervised: A training set is available containing only normal points and the task is to identify anomalous points in a test set. This is also called novelty detection:

$$\begin{aligned} \mathcal{D}_{train} &= X_{train} \sim p_{normal}(x) \\ \mathcal{D}_{test} &= X_{test} \sim p_{full}(x) \end{aligned}$$

- Unsupervised: A dataset containing both normal and anomalous points is available and the task is to identify the anomalous ones:

$$\mathcal{D} = X \sim p_{full}(x)$$

In this work, we focus on *unsupervised anomaly detection*. More specifically, given the full set of points $X \sim p_{full}(x)$, we want to find a subset $X_{anom} \subset X$ containing only anomalous points. The full distribution p_{full} is a mixture of distributions and it is a well-known result that general mixture models are unidentifiable [17], [18]. Thus, we should not expect to gain information on p_{anom} from knowing p_{full} for any small λ without a prior on the anomaly probability distribution, leading us to conclude that unsupervised anomaly detection requires a prior on p_{anom}.

Figure 1 shows a simple example where we illustrate a data distribution composed of three classes of points clustered in four visibly separable clusters. Anomaly detection is an undecidable problem under this setting without further information, since it is impossible to know if the dense cluster is composed of anomalies or the anomalies are the unclustered low density points (or even a combination of both). If we use a low capacity model, the cluster (Figure 1 on the Left) would probably present a higher anomaly score. If we use a high capacity model, the low density points (Figure 1 on the Right) would be detected as anomalous. Our choice of capacity implicitly imposes a prior on the detected anomalies.

III. Active Learning Models

The usual strategy to unsupervised anomaly detection is training a parameterized model $p_\theta(x)$ to capture the full data distribution $p_{full}(x)$. Also, since λ is by definition a small constant, it is typically assumed that $p_{full}(x) \approx p_{normal}(x)$. Finally, assuming points with low $p_{full}(x)$ values are anomalous [8], an anomaly score is usually defined to be $s(x) \propto \frac{1}{p_\theta(x)}$. There are three main issues with this strategy:

- if anomalies are more common than expected, p_{full} might be a poor approximation of p_{normal};
- if anomalies are tightly clustered in some way, high capacity models may learn to identify that cluster as a high probability region;
- if anomalies are as rare as expected, and since we only have access to p_{full}, we have no information about p_{anom} without further assumptions on its probability distribution.

Arguments for Active Learning: The aforementioned issues together argue in favor of an active learning strategy for anomaly detection, including auditor experts in the system's training loop. Thus, anticipating feedback and benefiting from it to find anomalies. Further, having an extremely unbalanced dataset ($\lambda \approx 0$) is another justification for adopting the active learning setting [19], [20], which has the potential of requiring exponentially less labeled data than in supervised settings [21]–[23].

A. The UAI Layer

Unsupervised anomaly detection, by itself, usually presents small accuracy in most practical scenarios, hence it is commonly used to rank instances which are later evaluated by human experts.

We consider here, then, the task in which we are given a dataset $\mathcal{D} = \{x | x \sim p_{full}(x)\}$, from which possibly anomalous data points are ranked and then sent to be audited by human experts, until a budget b is consumed.[1] If instead

[1] In these settings, it is not uncommon for large companies to be willing to label considerable sets of data ($b > 1000$ instances) in internal auditing processes.

of ranking and selecting all instances once, we iterate with experts in small batches (of k instances each), we can increase the number of anomalies found in these b labeled instances.

In our active learning setting, then, we iterate with experts. At each step, the $k \ll b$ data points most probably anomalous are sent to be audited and a new training regime takes place once the expert feedback returns.[2] This strategy of selecting the top k elements at each step is called most-likely positive. It is a common approach for selecting informative instances in highly imbalanced datasets [10], [24], and follows recent work in active anomaly detection [12]–[14].

With this in mind, we develop the UAI – Unsupervised to Active Inference layer. This layer can be incorporated on top of any unsupervised deep learning anomaly detection model which provides an anomaly score for ranking anomalies (e.g., a denoising auto-encoder). It takes as input both a latent representation layer ($l(x)$), created by the model, and its output anomaly score ($s(x)$), and passes it through a classifier to find an item's anomaly score, which is formally defined as:

$$\hat{p}(y|x) \propto s_{uai}(x) = classifier([l(x); s(x)])$$

where $\hat{p}(y|x)$ is our empirical estimate of the probability of point x being anomalous. This is motivated by recent work stating learned representations have a simpler statistical structure [25], which makes the task of modeling this manifold and detecting unnatural points much simpler [26]. In this work, we model the UAI layer using a simple logistic regression as our classifier, but any other classifier could be used as well. The classifier is thus given as:

$$\hat{p}(y|x) \propto s_{uai}(x) = \sigma(W_{act}[l(x); s(x)] + b_{act}) \qquad (1)$$

where $W_{act} \in \mathbb{R}^{1,d+1}$ is a linear transformation, $b_{act} \in \mathbb{R}$ is a bias term and $\sigma(\cdot)$ is the sigmoid function. We learn the values of W and b using back-propagation with a cross entropy loss function, where the targets are the few already actively labeled instances. We allow the gradients to flow through l, but not through s, since s might be non-differentiable. Hereafter we refer to networks that have an UAI layer as UAInets.

B. Architectures

Our models were obtained by incorporating the proposed UAI layer into two different anomaly detection architectures.

Denoising Autoencoder: The first model consists of a Denoising Auto-Encoder (DAE). Specifically, an encoder transforms the input into a latent space, and a decoder reconstructs the input using this latent representation. The loss function minimizes the reconstruction error L_2 norm. A denoising auto-encoder may be formally defined as follows:

$$\begin{aligned} l &= f_{enc}(x+\epsilon), \quad \epsilon \sim \mathcal{N}(0, \varphi) \\ \hat{x} &= f_{dec}(l) \\ \mathcal{L}_{DAE} &= ||x - \hat{x}||_2^2 \end{aligned}$$

[2]We should make it explicit here that this labeling process takes us out of the unsupervised anomaly detection setting. This is also not semi-supervised, though, since we start with no labels. This task is usually denoted as active anomaly detection [13], [14].

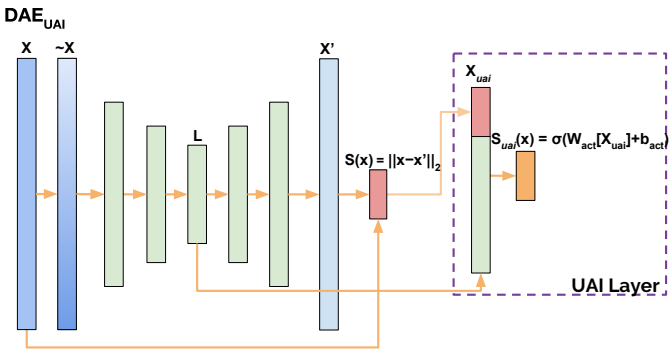


Fig. 2: A denoising auto-encoder with an UAI layer.

where both f_{enc} and f_{dec} are usually feed forward networks with the same number of layers, $l \in \mathbb{R}^d$ is a d-dimensional latent representation and ϵ is a zero mean noise, sampled from a Gaussian distribution with a φ standard deviation. When used in anomaly detection, the reconstruction error is used as an approximation for the anomaly score, as follows:

$$s_{DAE}(x) = ||x - \hat{x}||_2^2$$

Figure 2 presents a DAE_{uai} network which assembles the UAI layer on top of the autoencoder. This is formally defined as follows:

$$\begin{aligned} l_{DAE} &= l = f_{enc}(x+\epsilon) \\ s_{DAE_{uai}}(x) &= uai([l_{DAE}; s_{DAE}]) \end{aligned}$$

where $uai(\cdot)$ is simply the logistic classifier in Equation 1.

Multi-Layer Perceptron: Another unsupervised anomaly detection approach is training a multi-layer perceptron ($Class$) to predict an instance's class label (x_y) from its other features (x_x).[3] The cross-entropy of a data point serves as an estimate of its anomaly score, as follows:

$$\begin{aligned} \widehat{x}_y &= f_{Class}(x_x) \\ \mathcal{L}_{Class} &= H(x_y, \widehat{x}_y) = cross\_entropy\,(x_y, \widehat{x}_y) \\ s_{Class}(x) &= H(x_y, \widehat{x}_y) \end{aligned}$$

where $f_{Class}(\cdot)$ is a $p-$layer neural network.

Figure 3 presents $Class_{uai}$, which uses this classifier's last hidden layer (h_{p-1}) as a latent representation:

$$\begin{aligned} l_{Class} &= h_{p-1} \\ s_{Class_{uai}}(x) &= uai([l_{Class}; s_{Class}]) \end{aligned}$$

The full loss function used to train UAInets is:

$$\begin{aligned} \mathcal{L}_{uai} &= \mathbb{1}\ H(y, s_{uai}(x)) \\ \mathcal{L}_{full} &= \mathcal{L}_{uai} + \mathcal{L}_{base} \end{aligned}$$

where $\mathbb{1}$ has value 1 for already actively labeled instances, and 0 for unlabeled ones. $\mathcal{L}_{base}$ refers to the base network's loss function, which will be either $\mathcal{L}_{DAE}$ or $\mathcal{L}_{Class}$ here.

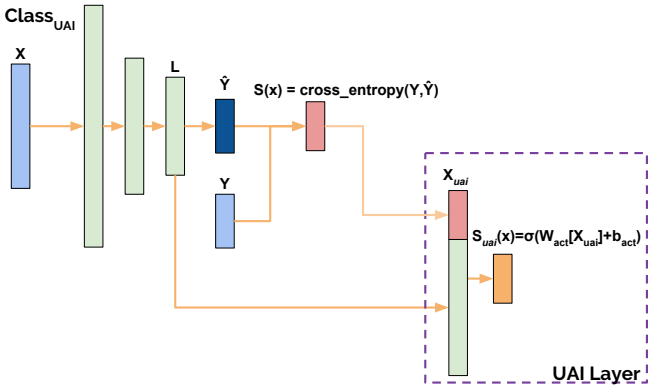


Fig. 3: A multi-layer perceptron with an UAI layer.

IV. Experiments

In this section, we evaluated our active learning models for anomaly detection. We assess the performance of the models by analyzing them on synthetic data created with different properties (Section IV-B) and on real anomaly data (Section IV-C).

A. Setup

For all experiments in this work the DAE's encoder and decoder have independent weights and we used both the DAE and $Class$ models with 3 hidden layers with sizes $[256, 64, 8]$. This means the latent representations provided to the UAI layers are $l \in \mathbb{R}^8$.

Algorithm 1 presents our Active Anomaly Detection process. We implemented all architectures using TensorFlow [27]. We set the learning rate to 0.01, batch size to 256 and we used the RMSprop optimizer with the default hyper-parameters.[4] We pre-trained the DAE and $Class$ models (by themselves, fully unsupervised and with no active sampling) for $5{,}000$ optimization steps. After that, for the active detection models, we select $k = 10$ data points to be labeled (i.e. normal or anomalous) at a time, and further train the full model for 100 iterations after each labeling call.

Algorithm 1 Active Anomaly Detection

procedure ActiveAnomalyDetection($\mathcal{D}$, expert, b, $k = 10$, $steps_{pre} = 5000$, $steps_{active} = 100$)
  $model$.pretrain($steps_{pre}, \mathcal{D}$)
  $i \leftarrow 0$
  $labels \leftarrow \emptyset$
  while $i < b$ **do**
    $model$.train($steps_{active}, \mathcal{D}, labels$)
    $top\_k \leftarrow model$.select_top($k, \mathcal{D}, labels$)
    $labels \leftarrow labels \cup expert$.audit($top\_k$)
    $i \leftarrow i + k$

[3]Note that even though class labels are available (x_y), we initially still have no anomaly label (y).

[4]Hyper-parameters were hand picked based on a few initial synthetic experiments on MNIST and not tuned in any further way.

B. Synthetic Data

When designing our experiments, we had the objective of showing that our model can work with different definitions of what is anomalous, while completely unsupervised models will need, by definition, to trade-off accuracy in one setting for accuracy in others.

MNIST Datasets: We used the MNIST dataset[5] and defined three sets of experiments:

1) **MNIST$_0$**: For the first set of experiments, we reduced the presence of the digits with the 0 label to only 10% of its original number of samples, making it only $1/91 \approx 1.1\%$ of the data points. The 0s still present in the dataset had their label randomly changed to $y \sim Uniform([1; 9])$ and were defined as anomalies.
2) **MNIST$_{0\text{-}2}$**: Follows the same dataset construction, but we reduce the number of instances of digits with labels 0, 1 and 2, changing their labels to $y \sim Uniform([3; 9])$, and again defining them as anomalous. Anomalies composed $3/73 \approx 4.1\%$ of the dataset.
3) **MNIST$_{\text{hard}}$**: This set of experiments aims to test a different type of anomaly. In order to create the corresponding dataset, we first trained a weak one hidden-layer classifier on MNIST and selected all misclassified data points as anomalous. Anomalies composed $\approx 3.3\%$ of the dataset.

Figure 4 presents results for these experiments and our main conclusion is that our models are robust to different types of anomalies, which is not the case for the unsupervised models. While $Class$ achieves good results in MNIST$_0$ and MNIST$_{0\text{-}2}$ datasets, it does not achieve the same performance in MNIST$_{\text{hard}}$, which might indicate it is better at finding clustered anomalies than low density ones. At the same time, DAE achieves good results for MNIST$_{\text{hard}}$, but performed poorly on MNIST$_0$ and MNIST$_{0\text{-}2}$, which indicates it is better at finding low density anomalies than clustered ones. Nevertheless, both UAInets are robust in all three experiments.

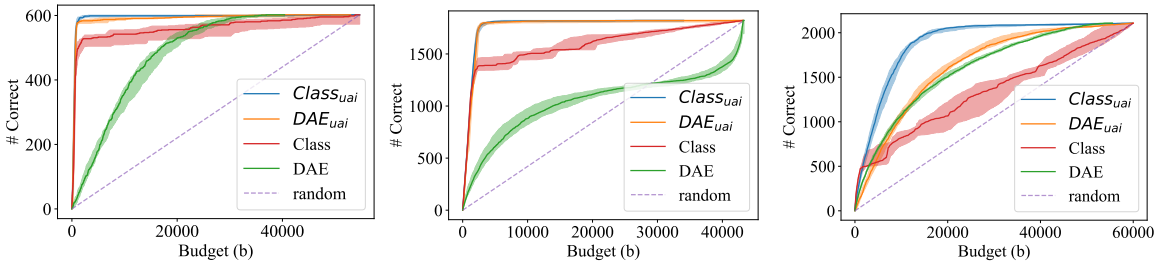
Fashion-MNIST Datasets: Table I presents statistics of datasets used to perform experiments on synthetic anomaly detection datasets based on Fashion-MNIST [29]. To create these datasets we follow the same procedures as with MNIST datasets, thus generating Fashion-MNIST$_0$, Fashion-MNIST$_{0\text{-}2}$, and Fashion-MNIST$_{\text{hard}}$.

TABLE I: Fashion-MNIST Anomaly Datasets.

	Dimension	# classes	# points	% anomalies
Fashion-MNIST$_0$	784	9	54,610	1.1%
Fashion-MNIST$_{0\text{-}2}$	784	7	43,765	4.0%
Fashion-MNIST$_{\text{hard}}$	784	10	60,000	16.1%

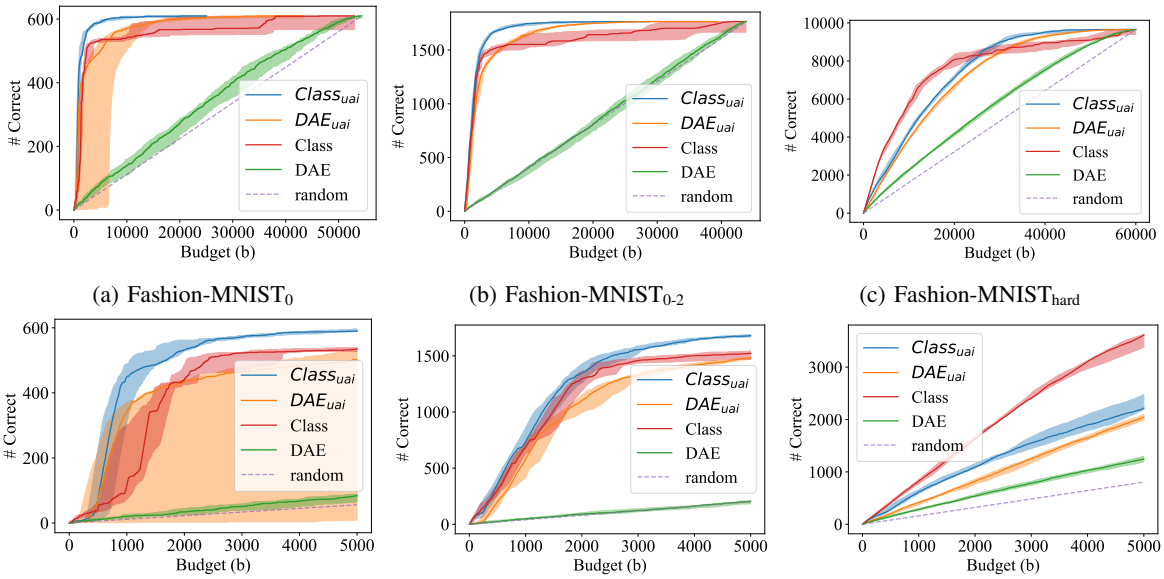
Figure 5 shows results of experiments performed on these datasets following the same procedures as with MNIST datasets. This figure shows similar trends to the ones for

[5]Using MNIST for the generation of synthetic anomaly detection datasets follows the same strategy as recent works [8], [28].



(a) MNIST_0 (b) $\text{MNIST}_{0\text{-}2}$ (c) $\text{MNIST}_{\text{hard}}$

Fig. 4: (Color online) Results for different MNIST experiments. Curves represent the average of five runs with different seeds. Confidence intervals represent max and min results for each budget b. y-axis represents number of anomalies detected with a specific budget. For MNIST_0, with a budget $b = 800$ we find almost all 600 anomalies with both UAInets, but less than 500 with $Class$ and only 30 with DAE.



(a) Fashion-MNIST_0 (b) Fashion-$\text{MNIST}_{0\text{-}2}$ (c) Fashion-$\text{MNIST}_{\text{hard}}$

(d) Fashion-MNIST_0 ($b < 5000$) (e) Fashion-$\text{MNIST}_{0\text{-}2}$ ($b < 5000$) (f) Fashion-$\text{MNIST}_{\text{hard}}$ ($b < 5000$)

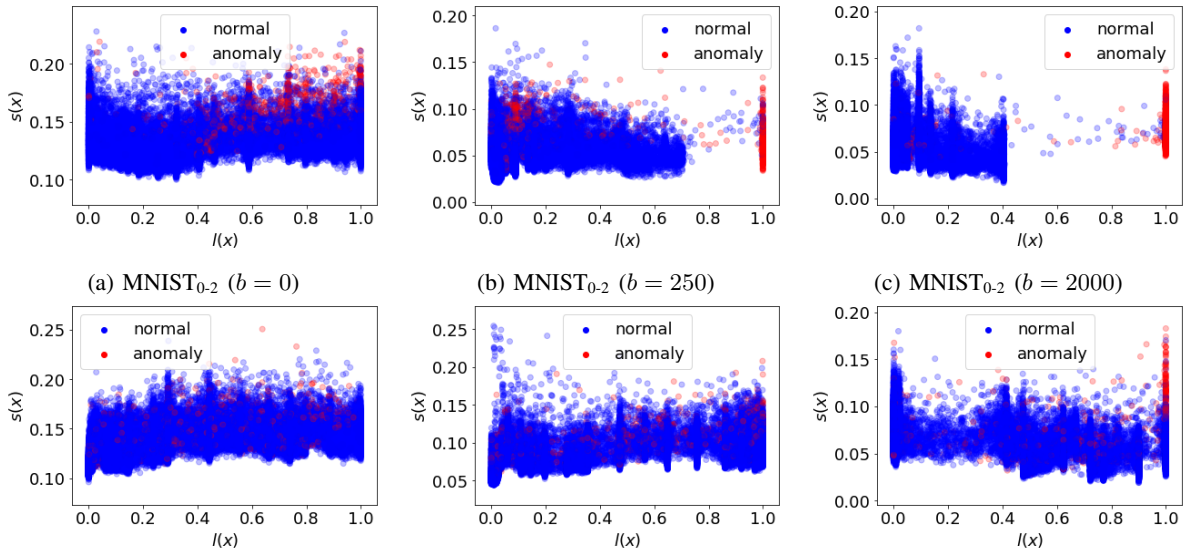
Fig. 5: (Color online) Results for Fashion-MNIST experiments, with different zooms on x-axis. Curves represent the average of five runs with different seeds. Confidence intervals represent max and min results for each budget b.

MNIST, although the anomalies in these datasets seem harder to identify. In one run of Fashion-MNIST_0, DAE_{uai} needed several active feedback iterations to start learning,[6] while for Fashion-$\text{MNIST}_{\text{hard}}$, $Class_{uai}$ takes a long time to start producing better results than $Class$. Nevertheless, UAInets are still much more robust than the underlying networks to different types of anomalies, producing good results in all datasets, even when the underlying network produces weak results on the dataset.

Learned Representations and Anomaly Scores: We further analyze UAInets by studying the evolution of hidden representations and anomaly scores through the training process. More specifically, we show visualizations of the learned representations and anomaly scores, presenting their evolution as more labels are fed into the network through the active feedback. With this purpose, we retrain our models on both $\text{MNIST}_{0\text{-}2}$ and $\text{MNIST}_{\text{hard}}$, with a hidden size of $[256, 64, 1]$, so that their latent representations are one dimensional ($l(x) \in R^1$), and plot these representations versus the anomaly scores of the base network (either DAE or $Class$) for different budgets (b).

Figure 6 shows the evolution of DAE_{uai} in terms of $l_{DAE}(x)$ and $s_{DAE}(x)$. We can see from Figures 6(a) and 6(d) that initially anomalies and normal data instances are not separable in this space. Nevertheless, with only a few labeled instances ($b = 250$) the anomaly space becomes much easier to separate, while for $b = 2000$ the anomaly space is almost perfectly linearly separable.

[6]Figures 4 and 5 confidence curves represent min and max results over five experiments with each model on each dataset. In Figure 5(d), in one of the experiments the autoencoders took a long time to find its first anomalous instance. It only found normal instances on its first 5000 picks, so the UAI layer had no signal from which to improve until then.



(a) MNIST$_{0\text{-}2}$ ($b = 0$) (b) MNIST$_{0\text{-}2}$ ($b = 250$) (c) MNIST$_{0\text{-}2}$ ($b = 2000$)

(d) MNIST$_{\text{hard}}$ ($b = 0$) (e) MNIST$_{\text{hard}}$ ($b = 250$) (f) MNIST$_{\text{hard}}$ ($b = 2000$)

Fig. 6: (Color online) Underlying latent representations (l_{DAE}) vs anomaly score (s_{DAE}) for DAE_{uai} network as training progresses.

Figure 7 shows the same evolution for $Class_{uai}$. We can also see the same patterns, as initially anomalies and normal data points are not separable, but with a few labeled instances anomalies become much more identifiable. The main conclusion taken from these visualizations is how the gradient flow through l is important, since it helps the network to better separate data in these anomaly spaces, allowing good anomaly detection performance even when the underlying networks are not good at identifying a specific type of anomaly.

C. Real Data

We also evaluated the performance of our models on publicly available anomaly detection benchmarks with real anomalies. These datasets were previously considered in [30]. We compare our model with DAE [31], DAGMM [6], LODA-AAD [13], and Tree-AAD [14].

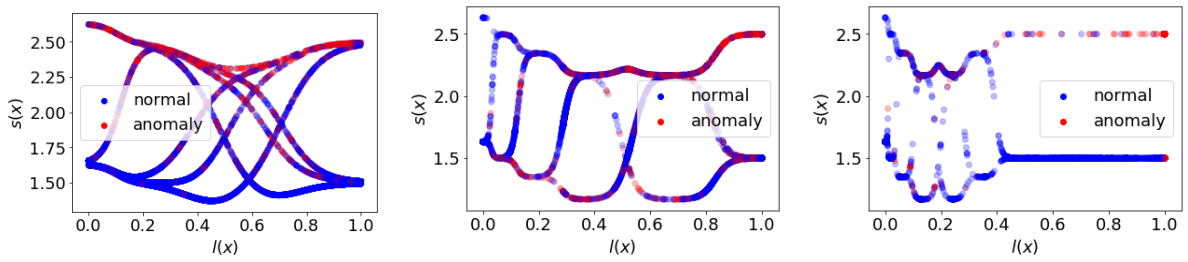
Table II presents results for these datasets. In these experiments, DAGMM (clean) was trained on a semi-supervised anomaly detection setting. DAGMM (dirty) and DAE were trained in an unsupervised setting. LODA-AAD, Tree-AAD and DAE_{uai} were trained in an active anomaly detection setting. DAE performs poorly in all the datasets. Nevertheless, even using a simple architecture as its underlying model, DAE_{uai} produces the best performance (or close to the best) on all datasets, even when the baselines were trained in completely clean training sets. DAE_{uai} also usually presents better results than LODA-AAD and Tree-AAD, which were also trained in an active setting.

One possible criticism would be that our results become more relevant the fewer the proportion of anomalous instances, which seems self-defeating. But we see that the largest difference from the active models to the other considered competitors was in Covtype, which has less than 1% anomalies out of 286,048 instances. When working with large datasets (>1M instances), even if only 0.1% of the dataset is contaminated there is still the chance to benefit from this feedback to improve performance. The active models are also more robust than the others, DAGMM used different hyperparameters for each experiment, while DAE_{uai} and AAD use the same for all (except for k which was reduced from 10 to 3 for the datasets with less than 100 anomalies).

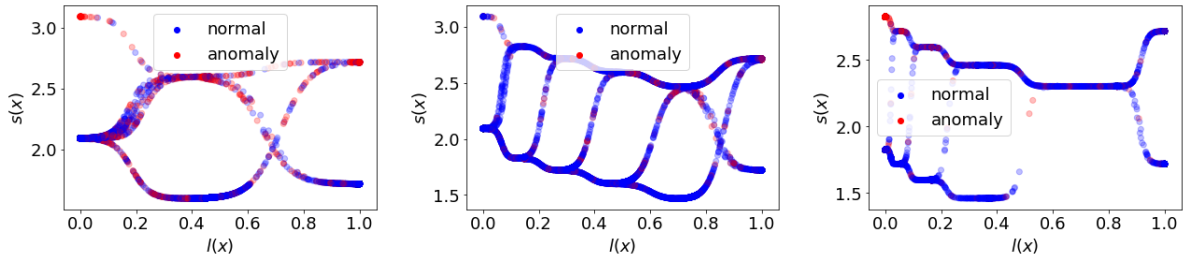
V. Related Work

Anomaly Detection: Although many algorithms have been recently proposed, classical methods for outlier detection like LOF [32] and OC-SVM [33], are still used in many application scenarios. Recent works have focused on statistical properties of "normal" data to identify anomalies, such as [34], which uses Benford's Law to identify anomalies in social networks, and [7], which uses Extreme Value Theory to detect anomalies. Recently, energy based models [28] and GANs [35] have been successfully used to detect anomalies, but autoencoders are still more popular in this field. The DAGMM algorithm was proposed in [6], where they train a deep autoencoder and use its latent representations, together with its reconstruction error, as input to a second network, which they use to predict the membership of each data instance to a mixture of gaussian models, training the whole model end-to-end in an semi-supervised manner for novelty detection.

Active Anomaly Detection: Over the years interesting works have been developed in this topic. In [36], the authors solve the rare-category detection problem by proposing an active learning strategy to datasets with extremely skewed distributions. In [37], the authors reduce outlier detection to classification using artificially generated examples that play



(a) MNIST$_{0\text{-}2}$ ($b = 0$) (b) MNIST$_{0\text{-}2}$ ($b = 250$) (c) MNIST$_{0\text{-}2}$ ($b = 2000$)



(d) MNIST$_{\text{hard}}$ ($b = 0$) (e) MNIST$_{\text{hard}}$ ($b = 250$) (f) MNIST$_{\text{hard}}$ ($b = 2000$)

Fig. 7: (Color online) Underlying latent representations (l_{class}) vs anomaly score (s_{class}) for $Class_{uai}$ network as training progresses.

TABLE II: Results on Real Datasets showing average F1 scores of five independent runs.

Train Set	KDDCUP	Arrhythmia	Thyroid	KDDCUP-Rev	Yeast	Abalone	CTG	Credit	Covtype	MMG	Shuttle
# Instances	494,021	3,772	452	121,597	1,191	1,920	1,700	284,807	286,048	11,183	12,345
# Features	120	6	274	120	8	9	22	30	54	6	9
% anomalies	20%	2.5%	15%	20%	4.6%	1.5%	2.6%	0.17%	0.9%	2.3%	7.0%
DAGMM (clean)	**0.94**	**0.50**	0.44	**0.94**	0.11	0.16	0.27	0.34	0.18	0.07	0.48
DAGMM (dirty)	0.43	0.46	0.46	0.31	0.02	0.05	0.18	0.31	0.01	0.00	0.48
LODA-AAD	0.88	0.45	0.51	0.83	0.31	0.54	0.52	0.57	**0.97**	0.42	**0.97**
Tree-AAD	0.89	0.29	**0.86**	0.50	0.32	0.53	**0.69**	**0.76**	0.94	0.59	0.92
DAE	0.39	0.35	0.09	0.16	0.23	0.08	0.13	0.36	0.15	0.27	0.17
DAE_{uai}	**0.94**	**0.47**	0.57	**0.91**	**0.33**	**0.55**	**0.66**	0.64	0.86	**0.60**	0.93

the role of potential outliers and then applies a selective sampling mechanism based on active learning to the reduced classification problem. In [38], the authors propose a Semi-Supervised Anomaly Detection (SSAD) method based on Support Vector Data Description (SVDD) [39]. In [12], the authors propose an active approach that combines unsupervised and supervised learning to select instances to be labeled by experts. In [10], they use an active learning approach to identify significant anomalies in aviation. They require explanations on expert annotators' choices, which they use to iteratively create new features with which they improve their model. The most similar works to ours in this setting are (i) [13], which proposes an algorithm that can be employed on top of any ensemble methods based on random projections, and (ii) [14], which expands Isolation Forests to work in an active setting.

VI. Conclusions and Future Work

In this work we proposed an Unsupervised to Active Inference layer (or simply UAI layer) that can be applied on top of any deep learning architecture designed for unsupervised anomaly detection. We showed that, even on top of very simple architectures like autoencoders and multi-layer perceptrons, our models achieve similar or better results than state-of-the-art representatives. To the best of our knowledge, this is the first work which applies deep learning to active anomaly detection. We used the strongest points of unsupervised deep learning solutions (learned representations and anomaly scores) to transform them into active learning models, presenting an end-to-end architecture which learns representations by leveraging both the full dataset and the already labeled instances.

Important future directions for this work are: (i) using the UAI layers confidence in its output to dynamically choose between either directly using its scores, or using the underlying unsupervised model's anomaly score to choose which instances to audit next; (ii) testing new architectures for UAI layers, in this work we restricted all our analysis to simple logistic regression; (iii) analyzing the robustness of UAINets to mistakes being made by the labeling experts; (iv) and making this model more interpretable, so that auditors could focus on a few "important" features when labeling anomalous instances, which could increase labeling speed and make their work easier.

REFERENCES

[1] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Art. Int. Review*, vol. 22, no. 2, pp. 85–126, 2004.

[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comp. Surveys*, vol. 41, no. 3, p. 15, 2009.

[3] C. C. Aggarwal, "Outlier analysis," in *Data mining*. Springer, 2015, pp. 237–263.

[4] N. Liu, X. Huang, and X. Hu, "Accelerated local anomaly detection via resolving attributed networks," in *IJCAI*. AAAI Press, 2017, pp. 2337–2343.

[5] G. Zheng, S. Brantley, T. Lauvaux, and Z. Li, "Contextual spatial outlier detection with metric learning," in *ACM SIGKDD Conf.* ACM, 2017, pp. 2161–2170.

[6] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *ICLR*, 2018.

[7] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouet, "Anomaly detection in streams with extreme value theory," in *ACM SIGKDD Conf.* ACM, 2017, pp. 1067–1075.

[8] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *ACM SIGKDD Conf.* ACM, 2017, pp. 665–674.

[9] B. Perozzi, L. Akoglu, P. Iglesias Sánchez, and E. Müller, "Focused clustering and outlier detection in large attributed graphs," in *ACM SIGKDD Conf.* ACM, 2014, pp. 1346–1355.

[10] M. Sharma, K. Das, M. Bilgic, B. Matthews, D. Nielsen, and N. Oza, "Active learning with rationales for identifying operationally significant anomalies in aviation," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016, pp. 209–225.

[11] J. F. Júnior, A. Pereira, W. M. Jr., and A. Veloso, "Methodology for fraud detection in electronic transactions," in *WebMedia Conf.*, 2012, pp. 289–292.

[12] K. Veeramachaneni, I. Arnaldo, V. Korrapati, C. Bassias, and K. Li, "Aiˆ 2: training a big data machine to defend," in *Big Data Sec. Conf.* IEEE, 2016, pp. 49–54.

[13] S. Das, W.-K. Wong, T. Dietterich, A. Fern, and A. Emmott, "Incorporating expert feedback into active anomaly discovery," in *ICDM*. IEEE, 2016, pp. 853–858.

[14] S. Das, W.-K. Wong, A. Fern, T. Dietterich, and M. Siddiqui, "Incorporating feedback into tree-based anomaly detection," *Workshop on Interactive Data Exploration and Analytics (IDEA)*, 2017.

[15] F. E. Grubbs, "Procedures for detecting outlying observations in samples," *Technometrics*, vol. 11, no. 1, pp. 1–21, 1969.

[16] D. M. Hawkins, *Identification of outliers*. Springer, 1980, vol. 11.

[17] B. Aragam, C. Dan, P. Ravikumar, and E. P. Xing, "Identifiability of nonparametric mixture models and bayes optimal clustering," *arXiv preprint arXiv:1802.04397*, 2018.

[18] L. Bordes, S. Mottelet, and P. Vandekerkhove, "Semiparametric estimation of two-component mixture model," *Annals of Statistics*, vol. 34, no. 3, pp. 1204–1232, 2006.

[19] R. Silva, M. Gonçalves, and A. Veloso, "A two-stage active learning method for learning to rank," *J. Assoc. Inf. Sci. Technol.*, vol. 65, no. 1, pp. 109–128, 2014.

[20] R. Silva, M. Goncalves, and A. Veloso, "Rule-based active sampling for learning to rank," in *ECML PKDD Conf.*, 2011, pp. 240–255.

[21] B. Settles, "Active learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6, no. 1, pp. 1–114, 2012.

[22] A. Ferreira, R. Silva, M. Gonçalves, A. Veloso, and A. H. F. Laender, "Active associative sampling for author name disambiguation," in *JCDL Conf.*, 2012, pp. 175–184.

[23] M. Moreira, J. dos Santos, and A. Veloso, "Learning to rank similar apparel styles with economically-efficient rule-based active learning," in *ACM ICMR Conf.*, 2014, pp. 361–370.

[24] M. Bilgic and P. N. Bennett, "Active query selection for learning rankers," in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2012, pp. 1033–1034.

[25] Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai, "Better mixing via deep representations," in *ICML*, 2013, pp. 552–560.

[26] A. Lamb, J. Binas, A. Goyal, D. Serdyuk, S. Subramanian, I. Mitliagkas, and Y. Bengio, "Fortified networks: Improving the robustness of deep networks by modeling the manifold of hidden representations," *arXiv preprint arXiv:1804.02485*, 2018.

[27] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. A. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning." in *OSDI Conf.*, vol. 16, 2016, pp. 265–283.

[28] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang, "Deep structured energy based models for anomaly detection," *arXiv preprint arXiv:1605.07717*, 2016.

[29] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," 2017.

[30] D. Dheeru and E. K. Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[31] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *ICML*. ACM, 2008, pp. 1096–1103.

[32] M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *SIGMOD Record*, vol. 29. ACM, 2000, pp. 93–104.

[33] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[34] S. Maurus and C. Plant, "Let's see your digits: Anomalous-state detection using benford's law," in *ACM SIGKDD Conf.* ACM, 2017, pp. 977–986.

[35] T. Schlegl, P. Seeböck, S. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *ICIPMI*. Springer, 2017, pp. 146–157.

[36] D. Pelleg and A. W. Moore, "Active learning for anomaly and rare-category detection," in *NeurIPS*, 2005, pp. 1073–1080.

[37] N. Abe, B. Zadrozny, and J. Langford, "Outlier detection by active learning," in *ACM SIGKDD Conf.* ACM, 2006, pp. 504–509.

[38] N. Görnitz, M. Kloft, K. Rieck, and U. Brefeld, "Toward supervised anomaly detection," *Journal of Artificial Intelligence Research*, vol. 46, pp. 235–262, 2013.

[39] D. M. Tax and R. P. Duin, "Support vector data description," *Machine learning*, vol. 54, no. 1, pp. 45–66, 2004.