Anomaly Detection

... reviewing machine learning literature about anomaly detection in fund (fund outliers)

A Unifying Review of Deep and Shallow Anomaly Detection

- Goal : identify the common underlying principles as well as the assumptions that are often made implicitly by various methods
- Link : https://research.google/pubs/pub50044/ (05/2021)
- Code : -
- Credible source: Google Research

Background knowledge

- anomaly : an anomaly is an observation that deviates considerably from some concept of normality
- type of anomalies
  1. point anomaly : an individual anomalous data point
     - i.e. illegal transaction in fraud detection or image of a damaged product in manufacturing
  2. conditional or contextual anomaly : anomalous data in a specific context such as time, space or the connections in a graph
     - i.e. price of $1 Apple stock might be normal back then, mean daily temperature below freezing point would be anomaly in Amazon
     - add contextual variable T in equation
  3. group or collective anomaly : set of related or dependent points that is anomalous
     - i.e. cluster of anomalies such as similar or related network attacks
- types of machine learning approaches for anomaly detection (shallow and deep learning anomaly detection) are usually referred to as low and high, which is the level in the feature hierarchy of some hierarchical distribution
  1. low-level / sensory anomalies
     - semantic concepts: individual characters and words
     - pixel-level features: edges or texture
  2. high-level / semantic anomalies
     - semantic concepts: topics
     - pixel-level features: objects and scenes in image
- anomaly, outlier, and novelty
  - anomalies are often the data points of interest (e.g., a longterm survivor of a disease)
  - outliers are frequently regarded as 'noise' or 'measurement error' that should be removed in a data preprocessing step ('outlier removal')
  - novelties are new observations that require models to be updated to the 'new normal'
- challenges in anomaly detection
  - mostly unsupervised nature of the problem
- different approaches
  - typical decision functions
    - one-class classification model : learn a discriminative decision boundary
    - probabilistic model : learn density
    - reconstruction model : learn some underlying geometric structure of the data
  - shallow and deep feature maps
    - one-class classification model : SVDD, Deep SVDD
    - probabilistic model : KDE, Flows
    - reconstruction model : PCA, AE
- probability density estimation (https://machinelearningmastery.com/probability-density-estimation/)
  1. the relationship between observations and their probability
  2. useful to know the probability density function for a sample of data in order to know whether a given observation is very unlikely as to be considered an anomaly
  3. parametric probability density estimation : involves selecting a common distribution (uniform, normal, exponential) and estimating the parameters for the density function from a data sample
  4. nonparametric probability density estimation : involves using a technique to fit a model to the arbitrary distribution of the data, like kernel density estimation
     a. still have parameters but not directly controllable in the same way as simple probability distributions
     b. i.e. using all observations in a random sample, in effect making all observations in the sample "parameters"

Model

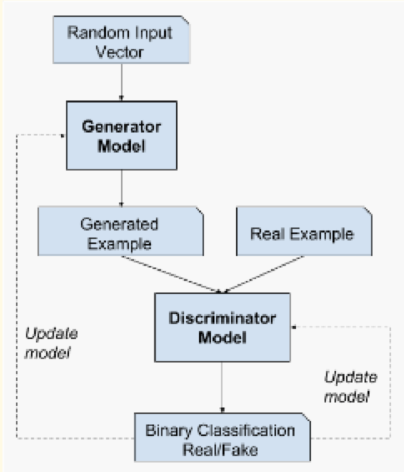
<table>
<tr><th colspan="2">approach</th><th>model</th><th>type</th><th>description</th><th>equation/pseudo code</th></tr>
<tr><td>density estimation and probabilistic models

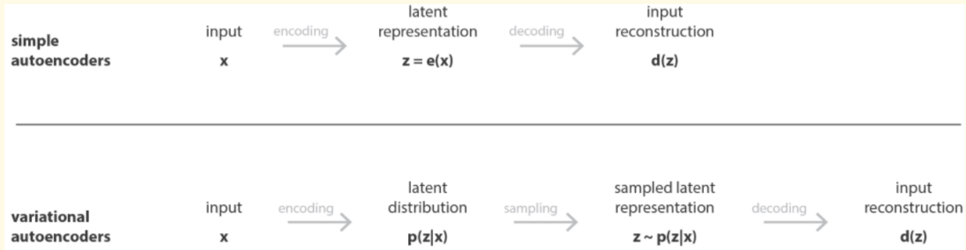
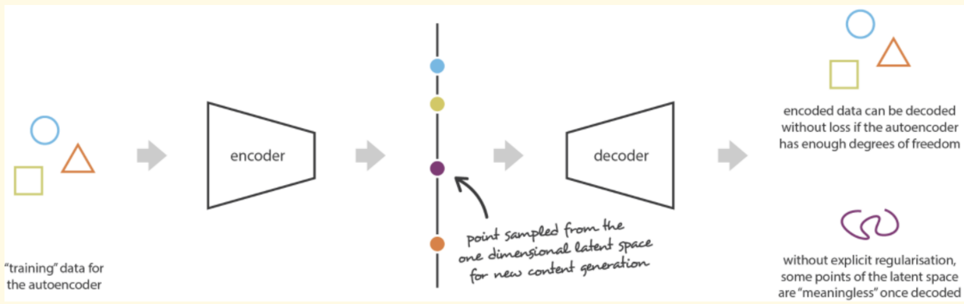
predict anomalies through estimation of the normal</td><td>Classic Density Estimation</td><td colspan="4">

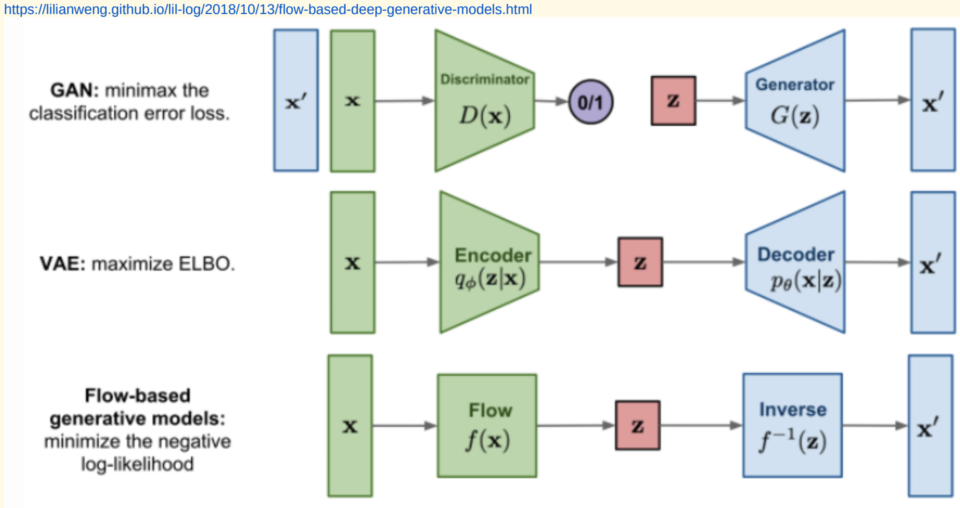
one of the most basic approaches to multivariate anomaly detection is to compute the Mahalanobis distance from a test point to the training data mean
Mahalanobis distance : a measure of the distance between a point P and a distribution D
i.e. KDE, histogram estimators, and Gaussian mixture models (GMMs)
classic nonparametric density estimators perform well for low dimensional problems, but they suffer from the curse of dimensionality
curse of dimensionality: the sample size required to attain a fixed level of accuracy grows exponentially in the dimension of the feature space.

</td></tr>
</table>

data probability distribution		Kernel Density Estimation (KDE)	baseline	• most widely used nonparametric density estimator • kernel is a mathematical function that returns a probability for a given value of a random variable • the kernel function weights the contribution of observations from a data sample based on their relationship to a given query sample • smooth parameter (bandwidth) : controls the number of samples or window of samples used to estimate the probability for a new point • https://mathisonian.github.io/kde/	$\hat{f}(x) = \sum_{observations} K(\frac{x - observation}{bandwidth})$
	Energy-Based Models	• a model whose density is characterised by an energy function • trained via gradient descent, and approximating the log-likelihood gradient via Markov chain Monte Carlo or Stochastic Gradient Langevin Dynamics • i.e. Deep Belief Networks and Deep Boltzmann Machines • not amenable to anomaly detection since one must marginalise out the latent variables to recover some value related to the likelihood			
	Neural Generative Models (VAEs and GANs)	• aim to learn a neural network that maps vectors sampled from a simple predefined source distribution • i.e. variational autoencoder(VAEs), and Generative Adversarial Networks (GANs)			

		Generative Adversarial Networks (GANs)	baseline	• a class of machine learning frameworks • consists of two neural networks, a generator network, and a discriminator network • two neural network contest with each other in a game (in the form of a zero-sum game, where one agent's gain is another agent's loss) • given a training set, this technique learns to generate new data with the same statistics as the training set • the core idea of a GAN is based on the "indirect" training through the discriminator, which itself is also being updated dynamically • this means that the generator is not trained to minimise the distance to a specific image, but rather to fool the discriminator. • in the end, we've trained the generative network to produce data that cannot be differentiated from real • this enables the model to learn in an unsupervised manner • http://kvfrans.com/generative-adversial-networks-explained/	 Example of the Generative Adversarial Network Model Architecture • implementation example : https://github.com/kvfrans/generative-adversial

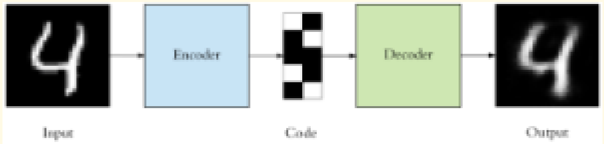
		Variational Auto Encoder (VAEs)	baseline	• variational auto encoder
				  ◦ an auto encoder whose encodings distribution is regularised during the training
				  ◦ in order to ensure that its latent space has good properties allowing us to generate some new data
				  ◦ add constraint on the encoding network that forces it to generate latent vectors that roughly follow a unit gaussian distribution
				  ◦ this constraint improves the generalisation of network
				  ◦ https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73
				• latent space : a representation of compressed data
					▪ variational auto encoder
					
					
					

		Normalising Flows (NFs)	advanced	• deep generative model which is better and more powerful distribution approximation
				• transforms a simple distribution into a complex one by applying a sequence of invertible transformation functions
				• unlike GANs and VAEs, the model explicitly learns the data distribution and therefore the loss function is simply the negative log-likelihood.
				• "normalising" : the change of variables gives a normalised density after applying an invertible transformation
				• "flow" : the invertible transformations can be composed with each other to create more complex invertible transformations
				• requirements
				  ◦ the input and output dimensions must be the same
				  ◦ the transformation must be invertible
					▪ difference between GAN, VAE, and flows
					  ▪ https://lilianweng.github.io/lil-log/2018/10/13/flow-based-deep-generative-models.html
					
					▪ Normalising Flows implementation
					  ▪ https://medium.com/swlh/normalizing-flows-are-not-magic-22752d0c924
					  ▪ https://github.com/pierresegonne/VINF

one-class classification models
 	
adopts a discriminative approach to anomaly detection
 	
used when we only have data of one class and the goal is to test new data and found out whether it is alike or not like the training data
 	
key question is how to minimise the miss rate for some given target false alarm rate with access to no anomalies
	Kernel-based One-Class Classification	One Class Support Vector Machines (OC-SVM)	baseline	• Support Vector Machines
				  ◦ supervised learning models for classification
				  ◦ find a hyperplane that separates the data in feature space with maximum margin from the origin
				• the difference from a standard SVM is that
				  ◦ it is fit in an unsupervised manner and
				  ◦ it provides a hyperparameter "*nu*" that controls the sensitivity of the support vectors and
				  ◦ should be tuned to the approximate ratio of outliers in the data, e.g. 0.01%
				• https://machinelearningmastery.com/one-class-classification-algorithms/
				• https://arshren.medium.com/anomaly-detection-techniques-part-2-92e1a7bc411c
					▪ SVM
					
					Data : Dataset with p^* variables and binary outcome.
					Output: Ranked list of variables according to their relevance.
					
					Find the optimal values for the tuning parameters of the SVM model;
					Train the SVM model;
					$p \leftarrow p^*$;
					while $p \geq 2$ **do**
					    $SVM_p \leftarrow$ SVM with the optimized tuning parameters for the p variables and observations in **Data**;
					    $w_p \leftarrow$ calculate weight vector of the SVM_p $(w_{p1}, \ldots, w_{pp})$;
					    $rank.criteria \leftarrow (w_{p1}^2, \ldots, w_{pp}^2)$;
					    $min.rank.criteria \leftarrow$ variable with lowest value in $rank.criteria$ vector;
					    Remove $min.rank.criteria$ from **Data**;
					    $Rank_p \leftarrow min.rank.criteria$;
					    p ← p - 1 ;
					end
					$Rank_1 \leftarrow$ variable in **Data** $\notin (Rank_2, \ldots, Rank_{p^*})$;
					return $(Rank_1, \ldots, Rank_{p^*})$

		Support Vector Data Description (SVDD)	baseline	• find a data-enclosing hypersphere of minimum volume • data description method that can give the target data set a spherically shaped description and be used to outlier detection or classification • https://github.com/iqiukp/SVDD	

	Deep One-Class Classification	Deep OC-SVM	advanced	▪ selecting kernels and hand-crafting relevant features can be challenging and quickly become impractical for complex data ▪ deep one-class classification methods aim to overcome these challenges by learning useful neural network feature maps ▪ deep OC-SVM variants and deep SVDD and employ a hypersphere model and linear model with explicit neural feature maps ▪ http://proceedings.mlr.press/v80/ruff18a.html ▪ https://github.com/lukasruff/Deep-SVDD-PyTorch	
		Deep SVDD	advanced		
reconstruction models learn a model that is optimised to well-reconstruct normal data instances, thereby aiming to detect anomalies by failing to accurately reconstruct them under the learned model	Principal Component Analysis (PCA)		baseline	• a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets • by transforming a large set of variables into a smaller one that still contains most of the information in the large set • reduce the number of variables of a data set, while preserving as much information as possible • https://builtin.com/data-science/step-step-explanation-principal-component-analysis	1: **procedure** PCA 2: Compute dot product matrix: $\mathbf{X}^T\mathbf{X} = \sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})^T(\mathbf{x}_i - \boldsymbol{\mu})$ 3: Eigenanalysis: $\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$ 4: Compute eigenvectors: $\mathbf{U} = \mathbf{X}\mathbf{V}\boldsymbol{\Lambda}^{-\frac{1}{2}}$ 5: Keep specific number of first components: $\mathbf{U_d} = [\mathbf{u}_1, \ldots, \mathbf{u}_d]$ 6: Compute d features: $\mathbf{Y} = \mathbf{U_d}^T\mathbf{X}$

	Auto encoders (AEs)		advanced	▪ learn deep latent variable models ▪ the aim of an autoencoder is to learn a representation for a set of data, typically for dimensionality reduction, by training the network to ignore signal “noise” ▪ compress the input into a lower-demensional 'code' and then reconstruct the output from this representation ▪ the code is a compact 'summary' or 'compression' of the input, called the 'latent-space representation' ▪ encoder compresses the input and produces the code ▪ decoder reconstructs the input only using this code ▪ to build auto encoder we need an encoding method, decoding method, and a loss function to compare the output with the target ▪ loss function : use mean squared error or binary crossentropy ▪ https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798 ▪ http://kvfrans.com/variational-autoencoders-explained/	▪ auto encoder  • implementation code : https://github.com/ardendertat/Applied-Deep-Learning-with-Keras/blob/master/notebooks/Part%203%20-%20Autoencoders.ipynb
	Prototypical Clustering - Vector Quantisation (VQ)	k-nearest neighbour algorithm (k-NN)	baseline	▪ assumes that similar things exist in close proximity ▪ https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761	1. Load the data 2. Initialise K to your chosen number of neighbours 3. For each example in the data a. Calculate the distance between the query example and the current example from the data b. Add the distance and the index of the example to an ordered collection 4. Sort the ordered collection of distances and indices from smallest to largest by the distances 5. Pick the first K entries from the sorted collection 6. Get the labels of the selected K entries 7. If regression, return the mean of the K labels 8. If classification, return the mode of the K labels

training data	experiment	method	output

1. Thyroid dataset (http://odds.cs.stonybrook.edu/) • includes n=3772 data instances and has D=6 real-valued features • contains a total of 93 (2.5%) anomalies	• our goal is to learn a model to detect thyroid gland dysfunctions such as hyperthyroidism • apply feature scaling to normalise value ranges	One-Class Support Vector Machine (OCSVM) with standard RBF kernel ▪ Radial Basis Function (RBF) kernel is commonly used in SVM	model achieves a test set AUC(Area Under the Curve) of 99.2%, a false alarm rate of 14.8%, and a miss rate of zero

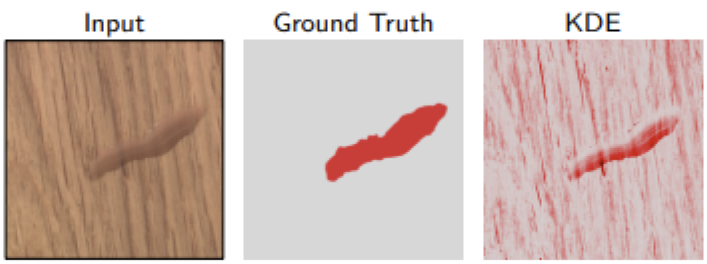
2. MVTec-AD dataset (https://www.mvtec.com/company/research/datasets/mvtec-ad)
• wood images
	• AUC compared with different models
	  ◦ Gaussian : 54.0
	  ◦ Minimum Volume Ellipsoids(MVE) : 80.1
	  ◦ Principal Component Analysis(PCA) : 90.4
	  ◦ Kernel Density Estimation(KDE) : 94.7
	  ◦ Support Vector Data Description(SVDD) : 94.1
	  ◦ Kernel PCA(kPCA) : 90.6
	  ◦ Generative Adversarial Networks(AGAN) : 74.5
	  ◦ Deep One-Class Classification(DOCC) : 91.6
	  ◦ AutoEncoder(AE) : 88.5
	• KDE does not compute higher-level image features such as DOCC
	  ◦ the anomalies involve properties such as small perforations and stains that do not require high-level semantic information to be detected
	  ◦ high AUC score must be due to a spurious correlation between the reaction of the model to stripes and anomalies
	• improve model
	  ◦ replace Gaussian kernel with Mahalanobis kernel
	  ◦ AUC drops to 87
		Kernel Density Estimation (KDE)	image represents anomaly prediction

Data

1. Thyroid dataset

type	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_medication	pregnant	sick	tumor	lithium	goitre	...	binaryClass
hypothyroid	72	M	f	f	f	f	f	f	f	f	...	P
hypothyroid	15	F	t	f	f	f	f	f	f	f	...	P
hypothyroid	24	M	f	f	f	f	f	f	f	f	...	N

2. MVTec-AD dataset



Detection of Accounting Anomalies in the Latent Space using Adversarial Autoencoder Neural Networks

- Goal : application of adversarial autoencoder networks that are capable of learning a semantic meaningful representation of real-world journal entries
- Link : https://arxiv.org/pdf/1908.00734.pdf (08/2019)
- Code : https://github.com/GitiHubi/deepAD
- Credible source: 2nd KDD Workshop on Anomaly Detection in Finance

training data	method	output

<table>
<tr>
<td>

real-world journal entries from SAP ERP system
synthetic dataset
<ol type="a">
data A - is an extract of an SAP ERP and encompasses the entire population of journal entries of a single fiscal year
data B - is an excerpt of the synthetic dataset presented in https://www.kaggle.com/ntnu-testimon/paysim1
pre-process the categorical entry to obtain a binary ("one-hot" encoded) representation by using pandas.get_dummies(), and numerical entry to be normalised
we inject a small fraction of synthetic global and local anomalies into both datasets.

one-hot encoding : each categorical level becomes a separate feature in the dataset containing binary values (1 or 0).

</td>
<td>
Adversarial Autoencoder Neural Networks

extends the concept of Autoencoder Neural Networks (AE) by imposing an arbitrary prior on the AEs latent space using a GAN training setup

</td>
<td>

global and local anomalies
global accounting anomalies are journal entries that exhibit unusual or rare individual attribute values. Such anomalies usually relate to skewed attributes, e.g., rarely used ledgers, or unusual posting times.
local accounting anomalies are journal entries that exhibit an unusual or rare combination of attribute values while their individual attribute values occur quite frequently, e.g., unusual combinations of general ledger accounts or user accounts used by several accounting departments.

</td>
</tr>
</table>

Data

BELNR	***WAERS***	***BUKRS***	***KTOSL***	***PRCTR***	***BSCHL***	***HKONT***	***DMBTR***	***WRBTR***	***label***
288203	*C3*	*C31*	*C9*	*C92*	*A3*	*B1*	*280979.6*	*0.0*	*regular*
324441	*C1*	*C18*	*C7*	*C76*	*A1*	*B2*	*129856.53*	*243343.0*	*regular*
133537	*C1*	*C19*	*C2*	*C20*	*A1*	*B3*	*957463.97*	*3183838.41*	*regular*

Model

<table>
<tr><th>type</th><th>model</th><th>description</th><th>equation/pseudo code</th></tr>
<tr>
<td rowspan="2">advanced</td>
<td rowspan="2">Autoencoder Neural Networks (AENs)</td>
<td>

a type of artificial neural network used to learn efficient data coding in an unsupervised manner
the aim of an autoencoder is

dimensionality reduction, by training the network to ignore signal “noise”.
discovering most important features

The number of input node and output node are same, and the number of hidden node is smaller.

</td>
<td>

```
input_size = 784
hidden_size = 128
code_size = 32

input_img = Input(shape=(input_size,))
hidden_1 = Dense(hidden_size, activation='relu')(input_img)
code = Dense(code_size, activation='relu')(hidden_1)
hidden_2 = Dense(hidden_size, activation='relu')(code)
output_img = Dense(input_size, activation='sigmoid')(hidden_2)

autoencoder = Model(input_img, output_img)
autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
autoencoder.fit(x_train, x_train, epochs=5)
```


input vector is 784 numbers between [0, 1]
128 nodes in the hidden layer
code size is 32
and binary crossentropy is the loss function.

</td>
</tr>
<tr>
<td colspan="2">

https://towardsdatascience.com/generating-images-with-autoencoders-77fd3a8dd368
https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

</td>
</tr>
</table>

<table>
<tr><td rowspan="4"></td><td rowspan="2">Variational Autoencoder (VAEs)</td><td>an autoencoder whose training is regularised to avoid overfitting and ensure that the latent space has good properties that enable generative process
objective: minimize reconstruction error + regularizer on latent space</td><td>

```
loss function {
        recon_loss = batch_size * MSE(input, decoder_output)
        kl_loss = - 0.5 * sum(1 + z_variance – square(z_mean) – exp(z_varaiance))
        total_loss = recon_loss + kl_loss
}

train function {
        input= sample_batch_data()
        z_mean, z_logvar = encoder(input)
        noise = sample_noise()
        z = z_mean + noise * z_variance
        decoder_output = decoder(z)
        loss = loss_fuction(input, decoder_output, z_mean, z_logvar)
        loss.backward()
}
```

</td></tr>
<tr><td colspan="2">https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73
http://kvfrans.com/variational-autoencoders-explained/</td></tr>
<tr><td rowspan="2">Generative Adversarial Networks (GANs)</td><td>a model architecture for training a generative model, and it is most common to use deep learning models in this architecture.
The GAN model architecture involves two sub-models: a generator model for generating new examples and a discriminator model for classifying whether generated examples are real, from the domain, or fake, generated by the generator model.
generator : model that is used to generate new plausible examples from the problem domain.
discriminator : model that is used to classify examples as real (from the domain) or fake (generated).</td><td>

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from noise prior $p_g(z)$.
- Sample minibatch of m examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from data generating distribution $p_{\text{data}}(x)$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(x^{(i)}\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)\right].$$

end for

- Sample minibatch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from noise prior $p_g(z)$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right).$$

end for

training drives the discriminator to attempt to learn to correctly classify samples as real or fake. Simultaneously, the generator attempts to fool the classifier into believing its samples are real. At convergence, the generator’s samples are indistinguishable from real data, and the discriminator outputs 1/2 everywhere. The discriminator may then be discarded.</td></tr>
<tr><td colspan="2">https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/
https://www.analyticsvidhya.com/blog/2017/06/introductory-generative-adversarial-networks-gans/</td></tr>
<tr><td>baseline</td><td colspan="3">N/A</td></tr>
</table>

Repository

topics	title (link)	goal	features	algorithm	output

anomaly detection	RLAD: Time Series Anomaly Detection through Reinforcement Learning and Active Learning • https://arxiv.org/abs/2104.00543	This paper introduces a new semi-supervised, time series anomaly detection algorithms that uses deep reinforcement learning and active learning.	• Yahoo Benchmark - Webscope dataset for time series anomaly detection. A1 Benchmark which shows the Yahoo membership login data and synthetic one, A2 Benchmark contains the anomalies of single outliers. • KPI dataset from AIOPs competition - collected from several internet companies such as Tencent, eBay and Sogou. There are 3,004,065 data points with timestamps and labels	• RLAD model • DRL(Deep Reinforcement Learning) • active learning - specific machines learning algorithm which allows the model actively interacts with user to obtain the desired learning experience. • reinforcement learning • deep q network • SR-CNN • Deep-SAD	

Anomaly detection

1. traditional approach
   a. simple machine learning algorithms based on distance and density
   b. usually time efficient but not able to achieve high performance
   c. KNN (K-Nearest Neighbourhood)
   d. LOF (Local Outlier Factor) > 1 – outlier
   e. LOCI (Local Correlation Integral)
   f. iForest (Isolation Forest)
   g. OCSVM (One-Class Support Vector Machine)
   h. PCA (Principle Component Analysis)
2. supervised approach
   a. Yahoo EGADS frameworks
   b. reply on tons of labels to train models
3. unsupervised approach
   a. not require labels and assume the abnormal points have larger deviation from the normal distribution
   b. the performance is low since they are difficult to leverage prior knowledge
   c. LinkedIn Luminol - light weight python library for time series data analysis
   d. Twitter TwitterAD
   e. DONUM - based on VAE (Variational Auto Encoder)
   f. Microsoft SR-CNN
   g. DAGMN - utilizes a deep auto encoder to generate a low-dimensional representation
4. semi-supervised approach
   a. uses a fraction of labeled data for training
   b. REPEN
   c. Deep-SAD
5. reinforcement learning approach
   a. Deep Q Network
      i. value based algorithm
      ii. agent learns the action value function and predicts how good to take an action as a specific state
      iii. unstable or even divergent when action value function is approximated with a nonlinear function like neural networks
   b. Active Learning
      i. allows the model actively interacts with user to obtain the desired learning experience
6. related technology
   a. statsmodelspython decomposition
   b. seasonal data - sarima model
   c. AR (Auto Regression) model
   d. point anomalies : an observation that deviates from the trend
   e. contextual anomalies : violation of seasonal trend
   f. collective anomalies : condition/contextual anomalies with respect to the entire data set

anomaly detection	Anomaly Detection in Turbofan Engine using Prophet • https://www.linkedin.com/pulse/anomaly-detection-turbofan-engine-using-prophet-apoorva-ravishankar?trk=public_profile_article_view • https://towardsdatascience.com/anomaly-detection-time-series-4c661f6f165f (simple example, and code is below link) • https://github.com/Diyago/ML-DL-scripts/tree/master/time%20series%20regression/anomaly%20detection • https://machinelearningmastery.com/time-series-forecasting-with-prophet-in-python/ (example with car sales data) • https://www.kaggle.com/vinayjaju/anomaly-detection-using-facebook-s-prophet (example with sunspot data) • https://docs.seldon.io/projects/alibi-detect/en/stable/examples/od_prophet_weather.html (example with weather data) • https://medium.com/swlh/facebook-prophet-426421f7e331 (explain methods)	We were able to effectively determine if the sensor data had any anomalous points.	• date and corresponding numeric value • sensor data collected from turbofan engines (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#turbofan)	• Prophet library (https://facebook.github.io/prophet/)	• a modular regression model with interpretable parameters • expected value, lower and upper boundary can be retrieved
anomaly detection	Interpretable, Multidimensional, Multimodal Anomaly Detection with Negative Sampling for Detection of Device Failure • https://research.google/pubs/pub49180 • http://proceedings.mlr.press/v119/sipple20a/sipple20a.pdf • https://github.com/google/madi	We propose an unsupervised anomaly detection method that creates a negative sample from the positive, observed sample, and trains a classifier to distinguish between positive and negative samples. We have demonstrated that negative sampling with random forest or neural network classifiers yield significantly higher AUC scores compared to state-of-the-art approaches against benchmark anomaly detection datasets, and a multidimensional, multimodal dataset from real climate control devices.	• We selected several familiar benchmark anomaly detection datasets from the Outlier Detection Dataset • The Smart Buildings anomaly dataset	• negative sampling : generate a negative sample from an unlabeled positive sample containing both normal and anomalous data, and then trains a classifier on the negative and positive samples to learn a decision boundary between normal and anomalous subspaces. • negative sampling classifiers: one using random forests and the other based on neural networks • Concentration of Measure Phenomenon : describes how manifolds distort as dimensionality increases, and is useful for characterizing anomaly detection algorithms in high-dimensional spaces, but has not been applied extensively in anomaly detection, with a few exceptions.	

fund category	Machine Learning Fund Categorisations • https://arxiv.org/abs/2006.00123	In this paper, we establish that an industry wide well-regarded categorisation system is learnable using machine learning and largely reproducible, and in turn constructing a truly data-driven categorisation.	• Morningstar Direct for one month • supervised learning	• decision tree • random forest • deep artificial neural network • compared to Morningstar categorisation	
fund group	Company classification using machine learning • https://arxiv.org/abs/2004.01496	In this paper, we demonstrate that unsupervised machine learning algorithms can be used to visualise and classify company data in an economically meaningful and effective way. The resulting company groups can then be utilised by experts in the field for empirical analysis and optimal decision making.	• daily returns of 318 companies from the S&P 500 index over a four-years, • Thomson Reuters industry classification system	• t-SNE • spectral clustering • PCA • unsupervised learning • validated by Thomson Reuter industrial code	
anomaly detection	Anomaly Detection in Univariate Time-series • https://arxiv.org/abs/2004.00433	This paper presents a quantitative comparison of multiple approaches on time-series data.	• Real Yahoo Services Network traffic • Synthetic Yahoo Service Network traffic • Synthetic Yahoo Service with Seasonality • Synthetic Yahoo Services with Changepoint Anomalies • NYC Taxi Dataset	• Statistical Approaches (AR-Model, MA-Model, ARIMA-Model, SES, ES, PCI) • Machine Learning Approaches (k-means, DBSCAN, LOF, isolation forest, One-Class SVM, XGBoosting) • Deep Learning approaches (MLP, CNN, WaveNet, LSTM, GRU, Autoencoder) • compared AUC-values and computation time in different approaches	
anomaly detection	Deep Reinforcement Learning for Unknown Anomaly Detection • https://arxiv.org/abs/2009.06847	This paper proposes an anomaly detection-oriented deep reinforcement learning approach that actively seeks and learns novel classes of anomaly that lie beyond the scope of the labeled anomaly data.	• NB15 - network intrusion datasets with a range of network attacks • Thyroid - dataset for detection of thyroid diseases • HAR - embedded inertial sensor data from a waist-mounted smartphone for six different human activities • Covertype - cartographic data of seven forest cover types	• Deep Q-Learning with Partially Labeled Anomalies (DPLAN) • deep reinforcement learning (DRL) • compared with five anomaly detectors (DevNet, Deep SAD, REPEN, iForest, DevNet+)	
anomaly detection	Time Series Anomaly Detection • https://research.google/pubs/pub46283/	Our goal is to utilise Machine Learning and statistical approaches to classify anomalous drops in periodic, but noisy, traffic patterns.	• google daily traffic	• DNN • RNN • LSTM • Deep Neural Network • Regression • TensorFlow	

https://paperswithcode.com/task/anomaly-detection/latest

https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00320-x

https://www.researchgate.net/publication/330364884_Recent_Progress_of_Anomaly_Detection

One major problem in building a predictive model for anomaly detection, is the scarcity of fraudulent data records in comparison to non-fraudulent, which makes the training data imbalanced.