Variational Autoencoders with Normalizing Flows for Unsupervised Anomaly Detection

Nihaar Shah [*] [1] Samuel Sharpe [*] [1]

1. Introduction

Anomalies represent deviations from an underlying process or distribution, and therefore, often reveal important information about novel events. Anomaly detection is an essential and difficult problem that has applications in a diverse set of industries and research areas such as fraud detection, network intrusion, industrial operations, healthcare, and social networks (Chalapathy & Chawla, 2019; Chandola et al., 2009).

Supervised methods for highly imbalanced data is often not possible, since anomalies by their definition are scarce. In cases where only ”normal” data is available, we must rely on unsupervised methods of anomaly detection. There are many existing unsupervised methods for anomaly detection including One-Class SVMs (Schölkopf et al., 2000), density based methods (Breunig et al., 2000; Knorr et al., 2000; Çelik et al., 2011), Isolation trees (Liu et al., 2008), variational autoencoders (An & Cho, 2015; Zimmerer et al., 2018; Lu & Xu, 2018; Xu et al., 2018), and generative adversarial networks (Yang et al., 2020; Zenati et al., 2018b;a).

2. Related Work

We focus on the use of variational inference, specifically variational autoencoders (VAEs) as a method of unsupervised anomaly detection. Flexibility of encoder/decoder architectures , stochastic elements, and easy training make VAE appealing for anomaly detection. VAEs have been used to detect outliers in tabular network data (An & Cho, 2015; Nguyen et al., 2019), image data (Zimmerer et al., 2018; An & Cho, 2015), and multi-modal data (Park et al., 2018).

Recently, there have been promising improvements of variational inference by applying non-linear invertible transformations, Normalizing Flows (NF), to samples from the prior to form more flexible, realistic, and multimodal posterior distributions (Rezende & Mohamed, 2015; Papamakarios et al., 2017; Kobyzev et al., 2019). However, applications of NF in anomaly detection to all types of data, but particularly tabular data has been limited. Ryzhikov et al. (2019) apply normalizing flows to supervised anomaly detection with novel loss functions for highly imbalanced data. Schmidt & Simic (2019) use recent advances in normalizing flows (Grathwohl et al., 2018) to detect anomalies from synthetic data and industrial time series data.

In this paper, we explore how normalizing flows can improve performance of variational autencoders for anomaly detection on tabular data.

3. Preliminaries

3.1. Variational Autoencoders

VAEs, like most generative models, assume that data samples are generated from an unobserved latent distribution. Formally, given i.i.d. data samples $\mathbf{x}_i \in \mathbf{X}$ and an unobserved latent variable $\mathbf{z}$, we assume $\mathbf{z}_i$ is drawn from some prior distribution $p_\theta\left(z\right)$. Subsequently, $\mathbf{x}_i$ is generated from a conditional distribution $p_\theta\left(\mathbf{x}|\mathbf{z}\right)$. Since these probability distributions are unknown, so are their parameters, θ.

Due to the lack of assumptions placed on these families of probability distributions, the marginal likelihood is often intractable. To circumvent the intractability of the estimate of $p_\theta\left(\mathbf{z}|\mathbf{x}\right)$, VAEs use a neural network as an approximation. This approximation, denoted $q_\phi\left(\mathbf{z}|\mathbf{x}\right)$, is called the *encoder*. Similarly, to map $\mathbf{z}$ to a distribution over $\mathbf{X}$, the probabilistic *decoder* is defined as $p_\theta\left(\mathbf{x}|\mathbf{z}\right)$. (Kingma & Welling, 2013).

Objective The VAE objective is derived from the Kullback-Leibler (KL) divergence between the approximation $q_\phi\left(\mathbf{z}|\mathbf{x}\right)$ to the actual posterior $p_\theta\left(\mathbf{z}|\mathbf{x}\right)$. We can expand the objective for one observation $\mathbf{x}_i$

$$\begin{aligned}\mathrm{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}_i\right) \parallel p_\theta\left(\mathbf{z}|\mathbf{x}_i\right)\right) = \log p_\theta\left(\mathbf{x}_i\right) + \\ \mathrm{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}_i\right) \parallel p_\theta\left(\mathbf{z}\right)\right) \\ - \ \mathbb{E}_{\mathbf{z}\sim q_\phi\left(\mathbf{z}|\mathbf{x}_i\right)}\left[\log p_\theta\left(\mathbf{x}_i|\mathbf{z}\right)\right]\end{aligned}$$

Since $\mathrm{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}_i\right) \parallel p_\theta\left(\mathbf{z}|\mathbf{x}_i\right)\right) \geq 0$, we have an expression for the lower bound on $\log p_\theta\left(\mathbf{x}_i\right)$ also known as the

[*] Equal contribution [1] Columbia University, New York, New York. Correspondence to: Nihaar Shah <ns3413@columbia.edu>, Samuel Sharpe <sbs2193@columbia.edu>.

evidence lower bound (ELBO):

$$\begin{aligned}\log p_\theta\left(\mathbf{x}_i\right) \geq & -\mathrm{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}_i\right) \| p_\theta\left(\mathbf{z}\right)\right) + \\ & \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\log p_\theta\left(\mathbf{x}_i|\mathbf{z}\right)\right] \\ & = \mathcal{L}(\theta, \phi, \mathbf{x}_i)\end{aligned}$$

Maximizing the ELBO is equivalent to the original objective since $\log p_\theta\left(\mathbf{x}\right)$ is constant with respect to $q_\phi\left(\mathbf{z}\right)$. Therefore, the parameters of the VAE encoder and decoder are found by solving

$$\arg\min_{\theta,\phi} -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\log p_\theta\left(\mathbf{x}_i|\mathbf{z}\right)\right] + \mathrm{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}_i\right) \| p_\theta\left(\mathbf{z}\right)\right)$$

Reparameterization In order to differentiate the objective and backpropogate with respect to ϕ, reparameterization of $q_\phi\left(\mathbf{z}|\mathbf{x}\right)$ is necessary. Instead of drawing $\mathbf{z}_i$ from $q_\phi\left(\mathbf{z}|\mathbf{x}_i\right)$, an independent random variable $\epsilon_\mathbf{i}$ is drawn from $p(\epsilon)$ and $\mathbf{z}_i$ is a deterministic transformation of $\epsilon_\mathbf{i}$:

$$\mathbf{z}_i = g(\epsilon_\mathbf{i}, \phi, \mathbf{x}_i)$$

In our experiments, we assume $q_\phi\left(\mathbf{z}|\mathbf{x}\right) = \mathcal{N}(\mu, \sigma^2\mathbf{I})$, so the reparameterization is simply

$$\mathbf{z}_i = \mu_i + \sigma \odot \epsilon_\mathbf{i}, \text{ where } \epsilon_\mathbf{i} \sim \mathcal{N}(0, \mathbf{I})$$

Objective Variations and KL-Annealing Research has shown that variations on the objective that constrain the regularization term, $\mathrm{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}\right) \| p_\theta\left(\mathbf{z}|\mathbf{x}\right)\right)$, lead to better disentangled latent factors (Higgins et al., 2017; Burgess et al., 2018). Additionally, techniques using warm up or annealing methods to change the influence of the regularization term have been shown to improve reconstruction ability and avoid KL collapse (Bowman et al., 2015; Burgess et al., 2018; Fu et al., 2019). We implement two KL annealing schemes for our experiments. Fu et al. (2019) use a cyclical KL annealing schedule of the with an objective formulation as seen in equation 1. Burgess et al. (2018) use a monotonically increasing schedule for their parameter C, which controls the effective encoding capacity of the bottleneck presented in equation 2.

$$\begin{aligned}\mathcal{L}(\theta, \phi, \mathbf{x}, \beta) = & \ \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta\left(\mathbf{x}|\mathbf{z}\right)\right] - \\ & \beta\, \mathrm{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}\right) \| p_\theta\left(\mathbf{z}\right)\right)\end{aligned} \tag{1}$$

$$\begin{aligned}\mathcal{L}(\theta, \phi, \mathbf{x}, C) = & \ \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta\left(\mathbf{x}|\mathbf{z}\right)\right] - \\ & \gamma|\mathrm{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}\right) \| p_\theta\left(\mathbf{z}\right)\right) - C|\end{aligned} \tag{2}$$

3.2. Normalizing Flows

Variational inference and therefore Variational Autoencoders have two primary challenges - both observable from the ELBO objective function discussed previously. Firstly, the efficient computation of $\nabla_\phi \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\log p_\theta\left(\mathbf{x}_i|\mathbf{z}\right)\right]$ and secondly choosing the richest yet feasible approximate posterior distribution $q(.)$. Typically VAEs assume an iid Gaussian approximate posterior $q_\phi(z|x)$ or other such mean field approximations. Consequently VAEs have limited flexibility to model the data if the true posterior $p_\theta(z|x)$ isn't coming from some such family of distributions. Normalizing flows are a means to remedy this inflexibility by allowing transformations of the base Gaussian distribution via learned parameters. By repeatedly applying a smooth, invertible mapping $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$ with inverse $f^{-1} = g$ we can transform a random variable $\mathbf{z} \sim q(\mathbf{z})$ into the resulting random variable $\mathbf{z}' = f(\mathbf{z})$ which has a distribution (Rezende & Mohamed, 2015)

$$q(\mathbf{z}') = q(\mathbf{z})|\det \frac{\partial f^{-1}}{\partial z'}| = q(\mathbf{z})|\det \frac{\partial f}{\partial z}|^{-1}$$

Combining a sequence of such transformations $\mathbf{z}_K = f_K o ... o f_1(\mathbf{z_0})$ we can find the log of the resulting distribution as:

$$\ln q_K(\mathbf{z}_K) = \ln q_0(\mathbf{z}_0) - \Sigma_k|\det \frac{\partial f_K}{\partial \mathbf{z}_{k-1}}| \tag{3}$$

These transformations' parameters are specified by neural networks. To make Normalizing flows feasible it is crucial to have an efficient means to evaluate the log-determinant of the Jacobian in the equation above. Typically computing a Jacobian of such transformations via a neural network has a complexity $O(LD^3)$ where L is the number of layers and D the dimension of the hidden layer. While choosing the NF it is thus imperative to keep in mind how to reduce this cost.

Planar flow is a particular transformation of z-space as follows where $h(.)$ is an element wise smooth non-linearity:

$$f(z) = z + u \times h(w^T z + b) \text{ with } u, w \in R^d \text{ and } b \in R \tag{4}$$

Let $\psi(z) = h(w^T z + b)$ in the equation above, then the following determinant is calculable due to the matrix-determinant lemma.

$$|\frac{\partial f}{\partial z}| = |1 + u^T \psi(z)| \tag{5}$$

Critically, for this mapping calculating the log-det of the Jacobian due to the above lemma is achievable in $O(D)$. This family of transformations is what we experiment with on our data. Also note that each planar flow is effectively modeling a single Multi-Layer Perceptron (MLP) type of transformation without the ability to have any information sharing across flows. This has its limitations of expressivity which we improved via another following flow.

3.3. Autoregression

To better scale flows to higher dimensions, we consider and use autoregressive autoencoders which introduce complex dependencies between the dimensions of the transformation. An autoregressive function has the following form (i being the dimension of $\mathbf{z}$):

$$z_i^{'} = f(z_{1:i})$$

Such functions do have a tractable jacobian-determinant because the Jacobian matrix $J = \frac{\partial z^{'}}{\partial z}$ will be lower triangular and so $\det J = \Pi_{i=1}^{D} J_{ii}$

Autoregressive transformation As discussed, we can connect a latent dimension z_i to all its lower dimensions $z_{1:i-1}$ for example let the base distribution $Q(z_0)$ be $z_0 \sim N(0, I)$ and its transformation be $z_0' = \mu_0 + \sigma_0 \odot z_0$. So the transformation is $z' = f(z)$. The other dimensions are then:

$$z_i' = \mu_i(z_{1:i-1}') + \sigma_i(z_{1:i-1}').z_i \tag{6}$$

The problem with this equation is computational complexity. The i-th dimension relies sequentially on i-1 dimensions all being *transformed* which is $O(D \times k)$ for D dimensions and k transformations. To improve on the complexity, consider the inverse of the equation (not fully derived here):

$$z_0 = \frac{z_0' - \mu_0}{\sigma_0} \tag{7}$$

$$z_i = \frac{z_i' - \mu(z_{1:i}')}{\sigma(z_{1:i-1}')} \tag{8}$$

Notice the that $\mu_i(.)$ has changed to $\mu(.)$. Due to the inversion $z' = f(z)$ has become $z = f(z')$ and it the inverse z_i now only depends on z' which in this case is the value *before* the transformation and so can be parallelized. Furthermore, the Jacobian determinant is lower triangular so we only need to compute the derivatives of the diagonal of $\frac{dz}{dz'}$ which is tractable.

3.4. Inverse Autoregressive Flows

Equation 8 contains $\mu(.)$ and $\sigma(.)$ which can be encoded by a neural network in addition to another parameter $\mathbf{h}$ which serves as an additional input to each subsequent step in the flow. (Kingma et al., 2016) So the autoregressive function is:

$$[\mu_t, \sigma_t] \leftarrow \text{autoregressive}[t](\mathbf{z_t}, \mathbf{h}; \theta) \tag{9}$$

Thus all the ingredients to compute distribution $Q(z_K)$ for performing variational inference with flows are available. The IAF equivalent for equation 3 sometimes also referred to as Variational Free Energy is:

$$E_{z \sim Q}[\log Q(z_0) - \Sigma_{k=1}^{K} \log|\det \frac{\partial z_t}{\partial z_{t-1}}| - \log P(x, z_K)] \tag{10}$$

Where we have defined z_t as a function of z_{t-1} in equation 8.

Algorithm 1 Anomaly score generation

```
Input: encoder $q_\phi(\mathbf{z}|\mathbf{x})$, decoder $p_\theta(\mathbf{x}|\mathbf{z})$, test data $X$
for $x_i \in X$ do
  $\mu_i, \sigma_i^2 = q_\phi(\mathbf{z}|\mathbf{x}_i)$
  Draw 100 samples $\mathbf{z}_{i,k} \sim \mathcal{N}(\mu_i, \sigma_i^2)$
  $A(x_i) = -\sum_{k=1}^{100} p_\theta(\mathbf{x}_i|\mathbf{z}_{i,k})$
end for
Rank $x_i$ descending in $A(x_i)$
Classify top n as anomalies
```

4. Methodology

We examine three types of models: a base VAE, VAE with planar flows, and VAE with IAR flows. For each of the experiments, we keep the architecture and training procedure in the VAE constant while changing the presence of normalizing flows.

We define our anomaly score $A(x)$ of an observation x to be the expected negative log likelihood:

$$A(x) = -\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]$$

This is proportional to the previously introduced concept of reconstruction probability (An & Cho, 2015). $A(x)$ will be high for samples with low reconstruction error (or likelihood). To determine anomalies, a threshold is often used as a cutoff for the value of $A(x)$. Since we know the distribution of classes in our data, we rank samples by $A(x)$ and classify the top X% as anomalies.

To estimate $A(x)$ we perform the steps in algorithm 1.

5. Data

We perform experiments on the 10% KDDCUP dataset downloaded from sklearn's api (Dua & Graff, 2017). The data, collected by DARPA, is composed of network connection statistics designed to help foster research into classifying network connections into normal and various intrusion types. As in other anomaly experiments (Yang et al., 2020), we use the intrusive classes as our ”normal” training samples and try to distinguish the non-attack connections at test time. Non-attack connections make up 20% of the dataset, so we rank our predictions by the anomaly score $A(x)$ and classify the top 20% as anomalies. We use 80% of the data for training and 20% for testing.

6. Experiments

We trained each model type five times with different random initializations. We measured performance using F1 score on the held out test data as shown in Table 1.

After exploring many different architectures and training parameters, We noticed that there were negligible changes with any amount of flows. Although our baseline VAE results are rather competitive to experiments in work by Yang et al. (2020) across different anomaly detection methods, we are surprised at the lack of improvement with normalizing flows and suspect there may be some flaws in the implementation.

We generated some synthetic circular 2D data (Figure 1) to diagnose issues in our implementation. We trained one type of each of the three different models on the synthetic data and plotted $A(x)$ in the data space (Figures 2, 3, 4) to visualize the underlying learned distribution.

We see that the base VAE and is unable to correctly capture the normal distribution of the circle. Using 20 layers of planar flows we are able to produce a distribution somewhat resembling the synthetic data, however, the results were sensitive to training and architecture parameters. We get even better results with IAF at 20 flow layers, almost resembling the true distribution.

Table 1. Performance (F1) of various models on hold out test set of KDDCUP dataset

Model	Flows	Mean F1	Stdev F1
Base VAE	N/A	0.938	0.0008
Planar VAE	5	0.936	0.0012
Planar VAE	50	0.937	0.0002
IAF VAE	5	0.936	0.0007
IAF VAE	50	0.911	0.0121

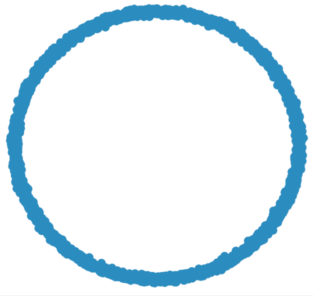


Figure 1. Synthetic data

7. Discussion Conclusion

While our results with our plain VAE are promising and competitive with other GAN based approaches, the addition of normalizing flows does not improve performance in our experiments. Aside from some possible implementation challenges, we do have some grounded theories that may explain some of the shortcomings of normalizing flows.

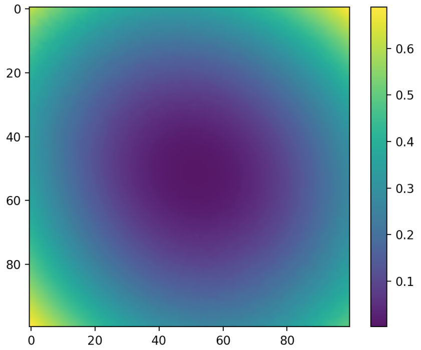


Figure 2. VAE: Predicted $A(x)$

The lack of improvement with planar flows may be due to their effectiveness in only relatively low-dimensional space (Kingma et al., 2016). We see in our visual exploration that planar flows does help model the latent distribution, but the results are sensitive to many of our parameters and was only partially successful with 20 stacked transformations in two dimensions.

While it was easy to explore different parameter settings in the two dimensional case, we lacked the time and computational resources to do extensive experiments on the KDD dataset. We hope and believe that there exist more appropriate latent dimension sizes and flow dimensions that will show improvements when using inverse autoregressive flows.

8. Project Notes

The code for the experiments is located here: `https://github.com/Nihaar1996/AnomalyDetection`.

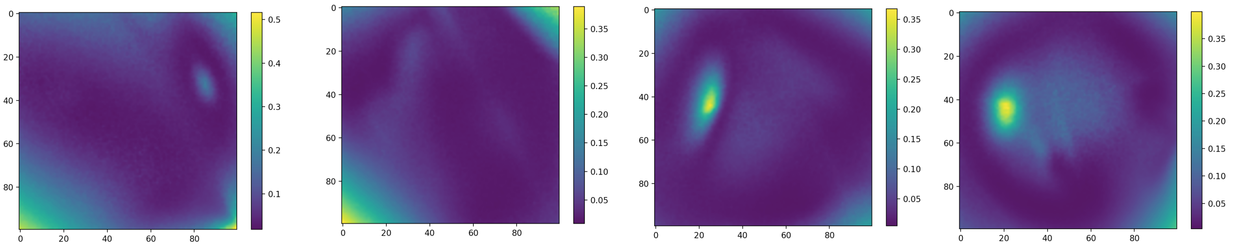
We originally proposed a different subject for our project and did some initial exploration here: `https://github.com/Nihaar1996/MetaClustering` . We decided to change directions later in the semester.

References

An, J. and Cho, S. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1), 2015.

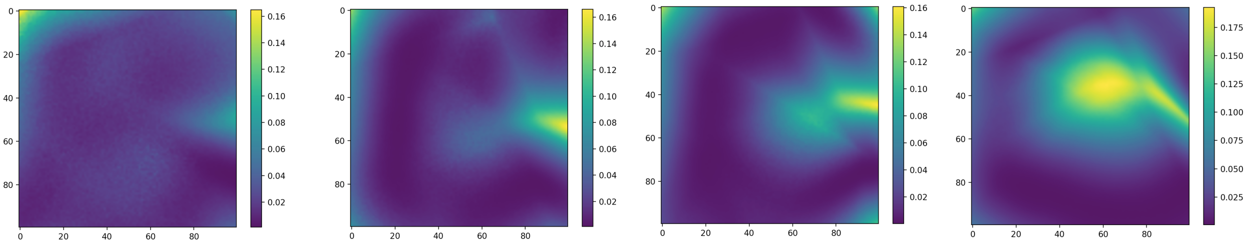
Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. Generating sentences from a continuous space, 2015.



(a) 5 planar flow layers (b) 10 planar flow layers (c) 15 planar flow layers (d) 20 planar flow layers

Figure 3. VAE with planar flow layers: Predicted $A(x)$



(a) 5 IAF Layers (b) 10 IAF Layers (c) 15 IAF Layers (d) 20 IAF Layers

Figure 4. VAE with IAF layers: Predicted $A(x)$

Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 93–104, 2000.

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., and Lerchner, A. Understanding disentangling in β-vae. *arXiv preprint arXiv:1804.03599*, 2018.

Chalapathy, R. and Chawla, S. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*, 2019.

Chandola, V., Banerjee, A., and Kumar, V. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3): 1–58, 2009.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Fu, H., Li, C., Liu, X., Gao, J., Celikyilmaz, A., and Carin, L. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. *arXiv preprint arXiv:1903.10145*, 2019.

Grathwohl, W., Chen, R. T. Q., Bettencourt, J., Sutskever, I., and Duvenaud, D. Ffjord: Free-form continuous dynamics for scalable reversible generative models, 2018.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework. *Iclr*, 2(5):6, 2017.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improved variational inference with inverse autoregressive flow. In *Advances in neural information processing systems*, pp. 4743–4751, 2016.

Knorr, E. M., Ng, R. T., and Tucakov, V. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3-4):237–253, 2000.

Kobyzev, I., Prince, S., and Brubaker, M. A. Normalizing flows: An introduction and review of current methods, 2019.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 413–422. IEEE, 2008.

Lu, Y. and Xu, P. Anomaly detection for skin disease images using variational autoencoder, 2018.

Nguyen, Q. P., Lim, K. W., Divakaran, D. M., Low, K. H., and Chan, M. C. Gee: A gradient-based explainable variational autoencoder for network anomaly detection, 2019.

Papamakarios, G., Pavlakou, T., and Murray, I. Masked autoregressive flow for density estimation, 2017.

Park, D., Hoshi, Y., and Kemp, C. C. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robotics and Automation Letters*, 3(3):1544–1551, 2018.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

Ryzhikov, A., Borisyak, M., Ustyuzhanin, A., and Derkach, D. Normalizing flows for deep anomaly detection, 2019.

Schmidt, M. and Simic, M. Normalizing flows for novelty detection in industrial time series data, 2019.

Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., and Platt, J. C. Support vector method for novelty detection. In *Advances in neural information processing systems*, pp. 582–588, 2000.

Xu, H., Feng, Y., Chen, J., Wang, Z., Qiao, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., and et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*, 2018. doi: 10.1145/3178876.3185996. URL http://dx.doi.org/10.1145/3178876.3185996.

Yang, Z., Bozchalooi, I. S., and Darve, E. Regularized cycle consistent generative adversarial network for anomaly detection. *arXiv preprint arXiv:2001.06591*, 2020.

Zenati, H., Foo, C. S., Lecouat, B., Manek, G., and Chandrasekhar, V. R. Efficient gan-based anomaly detection. *arXiv preprint arXiv:1802.06222*, 2018a.

Zenati, H., Romain, M., Foo, C.-S., Lecouat, B., and Chandrasekhar, V. Adversarially learned anomaly detection. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 727–736. IEEE, 2018b.

Zimmerer, D., Kohl, S. A., Petersen, J., Isensee, F., and Maier-Hein, K. H. Context-encoding variational autoencoder for unsupervised anomaly detection. *arXiv preprint arXiv:1812.05941*, 2018.

Çelik, M., Dadaşer-Çelik, F., and Dokuz, A. Anomaly detection in temperature data using dbscan algorithm. In *2011 International Symposium on Innovations in Intelligent Systems and Applications*, pp. 91–95, 2011.