Use of Uncertainty with Autoencoder Neural Networks for Anomaly Detection

Adrien Legrand, Harold Trannois, Alain Cournier

Use of Uncertainty with Autoencoder Neural Networks for Anomaly Detection.

Adrien Legrand*, Alain Cournier‡ and Harold Trannois§

Université de Picardie Jule Vernes
France
Email: *adrien.legrand@zenika.com, ‡harold.trannois@u-picardie.fr, §alain.cournier@u-picardie.fr

***Abstract*—Autoencoders neural networks are nonlinear dimension reduction models widely used in the field of anomaly detection. Conventionally, the reconstruction error is considered as a score function allowing the discrimination between the normal data and the outliers. Recent advances in calculating uncertainty from neural networks open new perspectives in the field of anomaly detection. We study, for given models and different concentrations of anomalies, several score functions. We compare the standard score function based on the standard error, a score based on the error resulting from the Bayesian approximation, as well as score functions directly including the uncertainty. This paper empirically demonstrates how including uncertainty in the score function is likely to improve the performance of an autoencoder-based anomaly detection model.**

***Index Terms*—Autoencoder Neural Network, Bayesian Neural Network, Prediction Uncertainty, Anomaly Detection**

I. Introduction

Anomaly detection finds extensive use in a wide variety of applications such as fraud detection for credit cards, insurance or health care, intrusion detection for cyber-security, fault detection... The importance of anomaly detection is due to the fact that anomalies in data often reflect critical exploitable information in a wide variety of areas. For example, an anomalous trafic pattern in a computer network could mean that a hacked computer is sending out data to an unauthorized destination [1]. In the medical field, abnormal images (MRI, X-rays...) may correspond to an illness or a injury [2]. Anomalies in credit card transaction data could indicate credit card or identity theft [3].

Anomalies are patterns in data that do not conform to a well defined notion of standard or normal behavior. The notion of normality purely depends on the overall tendencies present in our observation frame. Thus, anomalies are purely relative to the dataset they come from. Figure 1 illustrates various cases in a 2-dimensional data set. The data has a normal region N since most observations lie in it. Points sufficiently far away from the region are anomalies. The point a_1 is a clearly identified anomaly.

It is possible, however, that ambiguous data may be present within a dataset, making the anomaly detection task more delicate. In Figure 1, the point a_2 may or may not be considered as an anomaly, purely depending on a tolerance threshold that the analyst has to determine. In addition, we can also find a set of anomalies A that are close to each other. If new

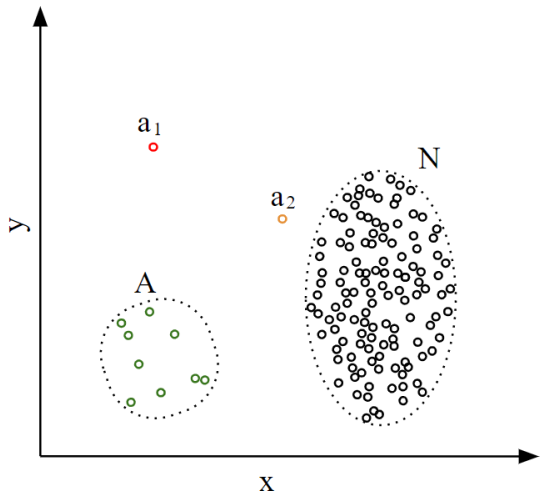

Fig. 1. Examples of normality, anomalies, and ambiguous data in a 2-dimensional dataset.

data were to be positioned near these anomalies, our notion of the normal behavior inherent in the basic dataset could be altered. Taking into account new data when using a model differentiates abnormality detection from novelty detection [4].

Anomalies might be induced in the data for a large variety of reasons closely related to the application domain where the dataset comes from. Malicious activity (e.g., credit card fraud [3], malware [5], terrorist activity, breakdown of a system [6]...), hardware malfunction (e.g., broken sensor, broken actuator [7]...), novelty [4] (one cause may be that the training dataset is undersized compared to the expected purpose)... but all of the reasons have a common characteristic that they are interesting to the analyst.

This paper aims to investigate a possible improvement in the performance of autoencoders-based anomaly detection systems, which are systems based on dimensionality reduction.

Dimensionality reduction has a lot of use cases applicable in many different areas [8]. Beside anomaly detection, dimensionality reduction facilitates the classification, visualization, communication, denoising,... of high dimensional data.

The reason that motivated the work from which is derived the autoencoder, is the principle of dimensionality reduction or feature learning.

Another example of a simple a widely used method of dimension reduction is Principal Components Analysis (PCA). PCA is a linear dimensionality reduction method which determinates directions of greatest variance in a dataset and project each data point along those news axes [9]. Although PCA is not the only algorithm (Robust PCA, Isomap, Diffusion maps... we advise the reader to refer to [10] for a complete review), there are many works that offer comparisons of various dimensionality reduction [11] [12] methods. Amongst those works, autoencoder neural appears, at least in some case, as a very viable alternative to other dimensionality reduction. According to the results in [13], autoencoders are among the most efficient models when the used datasets are real data.

Benefiting from the flexibility and adaptability of neural networks, autoencoders have the advantage of being able to efficiently use data which, with other dimensionality reduction methods, would require pretreatments or which would be more difficult to process. Thus, derived architectures have emerged, such as recurrent autoencoders [7] [14] used, for example, to reduce the size of time series or convolutional autoencoders [15] [16], which can extract visual features.

Those reasons makes autoencoder neural networks a significant and important part of anomaly detection algorithms.

Recent advances in the area of neural networks model uncertainty induced the emergence of processes, such as Monte Carlo Dropout [17] which we used in this paper. One of the purpose of prediction uncertainty is to capture the confidence of the model when producing a result. In this paper, we aim to integrate the results of recent work in this field to autoencoder based anomaly detection methods.

This paper makes the following contributions:

- Propose new score functions that include prediction uncertainty.
- Compare the performances of standard, recent and the previously mentioned new score functions.

The rest of this paper is organized as follows: Section II gives an overview of the background knowledge needed to fully understand the experimentation as well as related work about the use of autoencoder used in the context of anomaly detection, bayesian neural network, as well as model uncertainty. Section III explains how we defined new score functions. Section IV details the data we used for our experimentation, the methodology we followed, and the produced results. Section V concludes the paper.

II. Related Work

A. Anomaly detection with autoencoders

While classic neural networks are typically used for classification or regression problems, thus producing an output usually having a dimension different from the input, the main purpose of an autoencoder is to output a reconstruction of the input data [18].

Over training, an autoencoder neural network learns to approximate two functions. We consider here a dataset $\mathbf{X} = (\mathbf{x}, \ldots, \mathbf{x}_j, \ldots, \mathbf{x}_n)$ with $\mathbf{x}_j = (x_{j,1}, x_{j,2}, \ldots, x_{j,d}) \in \mathrm{R}^d$. The

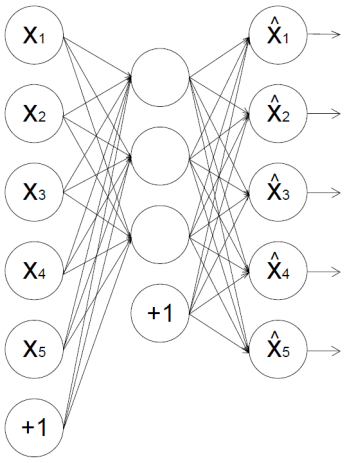

Fig. 2. A typical example of autoencoder.

encoding function f($\mathbf{x}$), for any $\mathbf{x} \in \mathbf{X}$, execute the dimension reduction and compress the data to a vector $\mathbf{h}$ with $\mathbf{h} \in \mathrm{R}^n$, n<d. The decoding function g($\mathbf{h}$) recreate an approximation of the original input $\hat{\boldsymbol{x}} \in \mathrm{R}^d$. We can see in Figure 2 a schema of a simple autoencoder with a single hidden layer. Given that n<d, the model is forced to prioritize which aspects of the input should be copied [19]. Autoencoder models are often restricted volontarily if they perform too well in order to learn the most useful properties of the data. In this way, during the test phase, the anomalies, which do not respect the learned properties of the data, will not be recreated with precision.

As shown by our experimentation in section IV, even if anomalies are present in the training dataset, the autoencoder will be more influenced the majority (i.e. normal data). Of course, the higher the anomaly concentration increases in the training game, the more the overall performance of the autoencoder to detect anomalies will tend to decline.

An important aspect for any anomaly detection technique is the manner in which the anomalies are reported [20]. Outputs produced by the majority of different anomaly detection methods can be divided into the following two types [21]:

a) Scores: Scoring techniques assign an anomaly score to each sample of the test data depending on the degree to which that instance is considered an anomaly. Thus the output of such techniques is a ranked list of anomalies. An analyst may choose to either analyze top few anomalies or use a cut-off threshold to select the anomalies. He therefore has the choice for ambiguous data (refer to Figure 1 for an example).

b) Labels: Techniques in this category assign a label (normal or anomalous) to each test instance. Scoring based anomaly detection techniques allow the analyst to use a domain specific threshold to select the most relevant anomalies. Techniques that provide binary labels to the test instances do not directly allow the analysts to make such a choice, though this can be controlled indirectly through

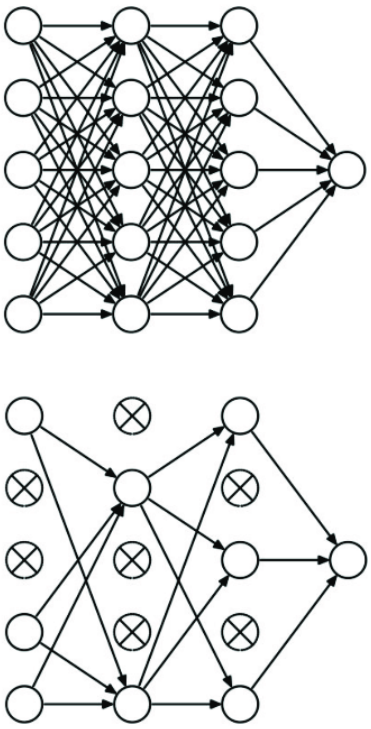

Fig. 3. Illustration of the principle of dropout applied to every layer of a neural network. Above is the original neural network and below is the same neural network with dropout applied.

parameter choices within each technique.

Autoencoders and other anomaly detection methods based on dimensionality reduction usually produces a score. This score is typically based on the distance between the reconstructed data from a compressed vector and the original data. The typical score function used with autoencoders is explained in Section III as Standard function (S.). An ideal autoencoder model will recreate a test dataset with a absolute discriminant error threshold, meaning that the anomaly with the smallest score will have a higher score than the normal sample with the highest score. Of course, this is rarely the case in reality, where we often have anomalous error distributions and normal error data that overlap. Generally, the choice that is made is to set the threshold in order to minimize the number of miscategorized anomalies, since they can be critical.

The performance of neural networks can be improved by applying dropout to the layers [22]. It has been shown that dropout prevents overfitting and provides a way of for neural network architectures to approximate functions more efficiently. Dropout can simply be considered as the allocation, for each neuron of a layer, of a probability of inactivity. The term "dropout" refers to dropping out units (hidden and visible) in a neural network, making them inactive. By "inactive", we mean temporarily removing it from the network, along with all its incoming and outgoing connections (Figure 3 illustrate how dropout applies to a neural network). The simplest case is, each unit is retained with a fixed probability p independent of other units, where p can be chosen using a validation set or can simply arbitrary set. Although work exists to find the best dropout rate for a model, the optimal dropout rate differs depending on the model. The trend is that larger models in terms of connections have a higher ideal dropout rate than smaller models.

Autoencoders may, in some cases, be capable of approximating the identity function. As mentioned above, the layer where the dimensionality reduction takes place is likely to be reduced if the autoencoder is too good at recreating the data. A more effective alternative is to incorporate dropout. Denoising autoencoders [23] consist of applying dropout to the input layer of the network to corrupt basic data during training. Thus, the network will never learn the identity function. In addition, work has shown that denoising autoencoders have a better ability to generalize and perform better in the context of anomaly detection [24].

All the models we used in our experiment are all inspired by autoencoding denoising by incorporating dropout on the input layers.

The Section II-B1 provides further details and more background about the architecture of the model used in the experiment.

B. Bayesian Neural Network & Uncertainty

1) Bayesian Neural Network: Bayesian neural networks (BNNs, Bayesian NNs) were first suggested in the '90s and have been studied in many works since then [25] [26]. Until recently, BNNs Bayesian neural networks were mostly theoritical model difficult to use on the field.

The purpose of BNNs if to offer a probabilistic interpretation of deep learning models by inferring distributions over the models' parameters rather than treating weights as simple real, 1-dimensional values. Such model would offer robustness to over-fitting, uncertainty estimates, and could easily learn from small datasets. A notable example of work using a Bayesian neural network is Uber, which has benefited from a significant performance increase through their implementation [27].

However, exact posterior inference is rarely possible. Indeed, BNNs are easy to formulate but difficult to perform inference with due to the complicated non-linearity and non-conjugacy in deep models. Such limitations motivated work on inference approximation.

Recently, several approximate inference methods are proposed for Bayesian Neural Networks. Most approaches are based on variational inference that optimizes the variational lower bound, including stochastic search [28], variational Bayes [29], probabilistic backpropagation [30], Bayes by BackProp [31] and its extension [32]. Several algorithms further extend the approximation framework to α-divergence optimization, including [33], [34]. We refer the readers to [35] for a more detailed and complete review of these methods. In all the previously mentionned algorithms, different training methods are required for the neural network. Various factors

and parameters must be adjusted, such as the loss function, which must answer to different optimization problems, and the training algorithm has to be modified in a usually non-trivial sense. However, in practice, solutions corresponding to the use case studied are often used, without changing the neural network architecture and can be directly applied to the previously trained model. In addition, most existing inference algorithms introduce additional model parameters, which makes the solution difficult to scale given the large amount of parameters.

In this paper, we use the same method as Uber [27], which is the Monte Carlo dropout (MC dropout). This method proposed in [17] and [36], which requires few or no change of the existing model architecture and provides uncertainty estimation almost for free. Specifically, stochastic dropouts are applied after each hidden layer in a classic neural network. The models we use are inspired by Denoising Autoencoders (cf: Section II-A), and thus already benefit from stochastic dropout applied on the input layer. The resulting architecture, for our case, is then an autoencoder with dropout applied to every layer. The output of our model can be approximately viewed as a random sample generated from the posterior predictive distribution [21]. As a result, the model uncertainty can be estimated by the sample variance of the model predictions in a few repetitions. In our cas, we will focus on prediction uncertainty rather than overall model uncertainty.

2) Bayesian and Uncertainty approximation: We consider here a neural network as function $f^{\boldsymbol{W}}$,where f denotes the function approximated by the model dependent on the set of parameters $\boldsymbol{W}$. Here, $\boldsymbol{W}$, representing the weights of each "layers" of connection accross the neural network, regroups weight matrices. Bayesian NNs often place a prior distribution over a neural network's weights, which induces a distribution over a parametric set of functions. Given weight matrix $\boldsymbol{W}_i$ and bias vectors $\boldsymbol{b}_i$ for layer i, a standard Gaussian prior distributions is often used:

$$p(\boldsymbol{W}_i) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$$

The model will then aims to fit the optimal posterior distribution through training.

We formulate the distribution of the data generated by the model by $p(y|f^{\boldsymbol{W}}(x))$. In the case of regression, it is often assumed

$$y|\boldsymbol{W} \sim \mathcal{N}(f^{\boldsymbol{W}}(x), \sigma^2) \tag{1}$$

with some noise level σ. In the case of classification, the softmax likelihood is often used. Autoencoders use multiple regression to recreate their input, so we will focus on the case of regression with the precision that $y = \hat{x}$, $\hat{x}$ being the approximation recreated for $f^{\boldsymbol{W}}(x)$.

Given a set of N observations $\mathbf{X} = (x_1, \ldots, x_n)$, Bayesian inference aims at finding the posterior distribution over model parameters $p(\boldsymbol{W}|\boldsymbol{X}, \hat{\boldsymbol{X}})$.

As previously explained, the model uncertainty can be determined by estimating $\text{Var}(\hat{\boldsymbol{X}}^*|\boldsymbol{X}^*)$, $\boldsymbol{X}^*$ a dataset of new data (i.e. test dataset), with Monte Carlo dropout. In our case, we will estimate prediction uncertainty by also using Monte Carlo dropout on each $x^* \in \boldsymbol{X}^*$ by determining $\text{Var}(f^{\boldsymbol{W}}(x^*))$.

The Monte Carlo dropout process is as follows: We consider an autoencoder $f^{\boldsymbol{W}}$, $\boldsymbol{W}$ having been determined by driving on an independent data set (note : in our case, dropout was also used during the training phase, cf Section II-A). For each new input x^* taken from a test dataset, we use $f^{\boldsymbol{W}}$ to compute a result $\hat{x}^*$ from it. The difference with the standard method comes from the fact that, during this passage in the network, random dropout is applied. We repeat this step B times, knowing that the inactivating neurons will not be the same each time, the dropout being random based on a probability p. We then obtain a set of results ($\hat{x}_1^*$,...,$\hat{x}_B^*$). From that set of approximations, we can determine two values that will be used in our score functions. The first is the "Bayesian approximation" which, if the number of iteration B is sufficient, will give a reliable average of the possible results w.r.t. 1 :

$$\bar{\hat{x}}^* = \frac{1}{B}\sum_{b=1}^{B} \hat{x}_B^* \tag{2}$$

The second is the prediction uncertainty, which is the variance of the distance between the result of each iteration of the Monte Carlo dropout and the Bayesian approximation (in our case, we used the squared difference) :

$$\text{Var}(f^{\boldsymbol{W}}(x^*)) = \frac{1}{B}\sum_{b=1}^{B} \left(\hat{x}_B^* - \bar{\hat{x}}^*\right)^2 \tag{3}$$

III. Studied score functions

A. Standard score functions

As detailed in II-A, autoencoders are part of anomaly detection algorithms that produce scores, essentially based on the distance between the input and the output. Moreover, as recalled in section II-B2, autoencoders work the same way as a multiple regression network. Thus, similar distance function can be used. In our case, we used the standard Mean Squared Error (MSE) distance function. Our first score function, which will now be called S. (Standard) for the rest of the paper, can be formulated as follow : For any sample $\boldsymbol{x} \in \mathrm{R}^d$

$$\text{Score}_S(\boldsymbol{x}) = \frac{1}{d}\sum_{i=1}^{n} \left(\boldsymbol{x}_i - f^{\boldsymbol{W}}(\boldsymbol{x})_i\right)^2 \tag{4}$$

B. Bayesian approximation based score functions

To our knowledge, no score function based on the Bayesian approximation has been studied in the context of anomaly detection in the literature. As we saw in section II-B2, the Bayesian approximation makes it possible to better take into account the distribution of the $\boldsymbol{W}$ parameters of the model. The result therefore benefits in part, theoretically, from the advantages induced by the Bayesian neural networks. It is then interesting to use the distance between this approximation and

the original data used as input. Our second score function, which will now be called B.A. (Bayesian Approximation based) for the rest of the paper, can be formulated as follow :

$$\text{Score}_{BA}(\boldsymbol{x}) = \frac{1}{d}\sum_{i=1}^{n}\left(\boldsymbol{x}_i - \bar{\hat{\boldsymbol{x}}}_i\right)^2 \quad (5)$$

with $\bar{\hat{\boldsymbol{x}}}$ being the Bayesian approximation explained by 2. It should be noted that, in order to use this score function, Monte Carlo dropout has to be implemented. Therefore, if one came to use it, it would be trivial to use the following proposed functions as well.

C. Score functions weighted by uncertainty

In order to increase the performance of autoencoders in the context of anomaly detection, any factor increasing the score of an abnormality or decreasing the score of a normal data can be taken into account. In order to develop the score functions that will follow, we formulate the following hypothesis:

- An anomaly, if it is sufficiently rare in the training data set and sufficiently different from the normal data (i.e. can be unambiguously designated as an anomaly, as explained in the Introduction), will have a weak impact on the distributions of the model parameters $\boldsymbol{W}$ post-training. Therefore, an anomaly, when passing through the network during the test phase, should have an associated variance $\text{Var}(f^{\boldsymbol{W}}(x^*))$ (explained in 3), and therefore a prediction uncertainty, higher than those associated to normal data.

However, we do not propose to use only prediction uncertainty as a score function. Indeed, our study shows that the hypothesis formulated is not always true and depends on factors over which we have little or no influence. The consequence is that, although uncertainty alone could be a good score function for some models, the uncertainty and standard error of a model are often not correlated. Indeed, a model is likely to give an estimate $\hat{x}$ quite far from x (x being a new anomalic data in that case) with relatively high certainty and vice versa for normal data. We therefore propose to study the impact if we enrich classical function score, based on the distance between the recreated data and the original input data, with the prediction uncertainty.

This observation guided us in setting up the functions: simply weighting the reconstruction error would have been a source of bad categorization of the anomalies. Indeed, an anomaly associated with a high error but with an uncertainty close to 0 would have had a final score close to 0. That is why we propose to weight the initial and previously described score functions by the uncertainty plus one. In this way, we seek to value the best of both approaches. The first weighted score functions, which will now be called W.S. (Weighted Standard) for the rest of the paper, can be formulated as follow :

$$\text{Score}_{WS}(\boldsymbol{x}) = \text{Score}_{S}(\boldsymbol{x}) * (1 + \frac{\text{Var}(f^{\boldsymbol{W}}(\boldsymbol{x}^*))}{reg}) \quad (6)$$

The term reg is used to put the uncertainty factor on the same scale as the error factor (here Score_S) in order to give them both the same importance on the final score.

The last weighted score functions, which will now be called W.B.A. (Weighted Bayesian Approximation) for the rest of the paper, can be formulated as follow :

$$\text{Score}_{WBA}(\boldsymbol{x}) = \text{Score}_{BA}(\boldsymbol{x}) * (1 + \frac{\text{Var}(f^{\boldsymbol{W}}(\boldsymbol{x}^*))}{reg}) \quad (7)$$

The motivation behind this score function is to use both the Bayesian approximation and the uncertainty to discriminate anomalies and normal data.

IV. Experimentation

To create, train, and test our model, we used the python implementation of the wrapper Keras with a tensorflow back-end. Every treatment made on the data were done using the libraries numpy and scikit-learn. The running environment of all the experiment was a Linux Virtual Machine equipped with a Nvidia Tesla V100 GPU and hosted on google compute engine.

A. Datasets

To evaluate our functions, we searched for datasets fulfilling certain criteria. Since autoencoders are based on dimension reduction, we have selected datasets with at least 15 features. We have not followed an upper limit in the number of features, so we have a set of varied datasets ranging from 16 features to 617.

The selected datasets are generally used for either multi-class classification or binary classification. We have, in the case of datasets used for multi-class classification, selected and merged several classes after having downsampled them to create our set of anomalies. In the case of datasets used for the binary classification, we have downsampled one of the two classes.

With this method of using classification data for evaluation of anomaly detection methods we are conform with the literature [37]. The datasets with a binary class are Musk [38], Wisconsin Breast Cancer (WBC) [39] and Ionosphere [40]. The other datasets we used are Low Resolution Spectrometer (LRS)[41], Isolet [42], Satimage [43], and Letter Recognition [44].

We therefore have a heterogeneous set of datasets, coming from various fields of application, having different dimensions and numbers of samples. It should be noted, however, that these datasets allow the use of a "standard" autoencoders (detailed in Section II-A). So we did not focus our study on datasets of images, videos, time series, ... and did not need architectures derived from autoencoders such as recurrent or convolutional autoencoders.

Those datasets are real data coming from the UCI machine learning repository[45] and have been used previously in the litterature in works about anomaly detection or classification. The details of the data and the anomaly classes are detailed in the table IV-A.

TABLE I
SUMMARY OF THE DIFFERENT DATASETS USED. ("NUMBER OF SAMPLES" IS APPROXIMATE BECAUSE IT REFERS TO THE SIZE OF THE TRAINING DATASET, WHICH VARIED ACCORDING TO THE ANOMALY CONCENTRATION USED.)

Dataset	Nb of features	∼ nb of samples	Anomaly class
LRS	101	360	classes 4 & 8
Isolet	617	7000	classes 23 & 26
Satimage	37	4000	class 2
Ionosphere	34	200	class 'bad'
WBC	30	320	class 'malign'
Musk	166	1100	class 'musk'
Letter Recognition	16	10000	classes 'M' & 'W'

B. Methodology

To evaluate our score functions, we have, for each dataset, tested models using training sets containing concentrations of anomalies that we have varied. In this way, we can better evaluate the different score functions since the challenge level of the anomaly detection task increases with the number of anomalies present in the training dataset.

Because of the different characteristics of the original datasets, we have not been able to use identical concentrations for each. In addition, we had to adapt the concentrations in anomalies to the capacities of the autoencoders used. For example, the score functions tested resulting from the autoencoder trained with the Ionosphere dataset with an anomaly concentration of 18.8% are very efficient, whereas those resulting from the best autoencoder trained with the LRS dataset with a concentration of anomalies at 2.2% are much less so.

For each given concentration of anomalies, we trained and tested several models with the same parameters. Indeed, the smaller the volume or the smaller the dimension of the used dataset, the more two models with the same parameters can produce different results due, for example, to the random nature of the dropout. In order not to bias the experiment, we repeated the trainings in order to look for the models producing the best standard score function (S.).

In order to be fair in our evaluation of the models and associated score functions, we use the same number of anomalies and normal data during the test phase. This allows us to use a single Receiver Operating Characteristic (ROC) curve measurement [46] for each model (each anomaly concentration for each dataset) while not underestimating the value of the anomalies due to their rare nature. This method is widely used amongst the literature to evaluate model performances.

C. Results

We recall that our initial hypothesis is that the prediction uncertainty, estimated with $\mathrm{Var}(f^{\boldsymbol{W}}(x^*))$, is higher when x^* is an anomaly rather than when it is normal (as explained in section II-B2). Although this hypothesis is not always verified, the cases where it is true allow to have more efficient function score than the classic function score.

We find that :

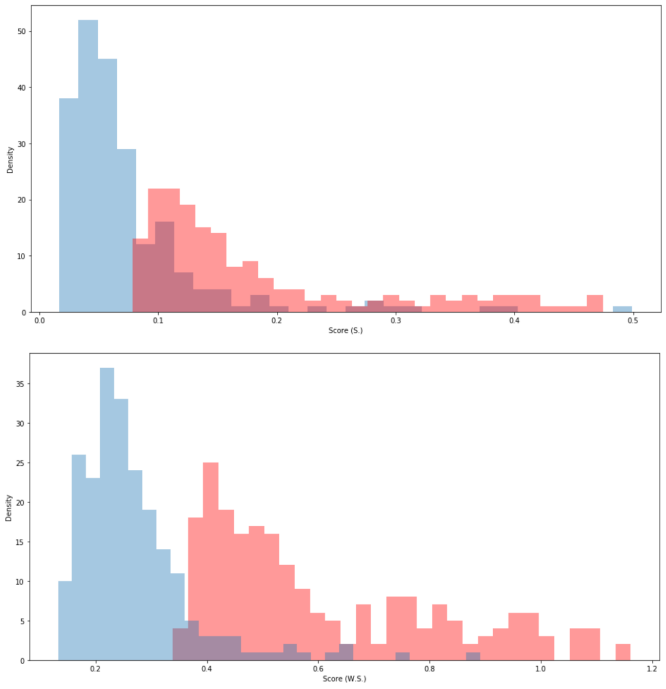

Fig. 4. Display of the distribution of the anomaly set scores (in red), the normal data set scores (in blue) for the Standard (S. above) function and the Weighted Standard (W.S. below) function applied to the Satimage test dataset, with a training dataset including 2.9% anomalies.

- The standard function score (S.) is the best in only two cases (∼9.5%).
- The function score based on the Bayesian approximation (B.A.) is the best in 6 cases (∼28.5%).
- The function score using the basic error and uncertainty (W.S.) is the best in seven cases (∼33.3%).
- The function score using Bayesian approximation and uncertainty (W.B.A.) is the best in 6 cases (∼28.5%).

We can consider that the score functions not using uncertainty and those using it form two categories. For a given model, the scores within each category are, for the most part, similar. We deduce that, generally, it is the consideration of uncertainty is the determining factor in the increase or decrease of performance, depending on the models. The category including prediction uncertainty seems to have overall better performances. Figure 4 illustrates an example, where we can see that the section size where the normal distribution and the anomaly distribution overlap is smaller when the W.S function is used, compared to the S function.

The form of the data used (dimension, quantity) does not seem to determine which score function will be the best. Nevertheless, it seems that the general ability of an autoencoder to discriminate anomalies is a good indicator. We can see on the figure 5 that the category not using uncertainty has generally higher best scores. The use of uncertainty tends to lower the performance of the score function when the autoencoder has already very good performances.

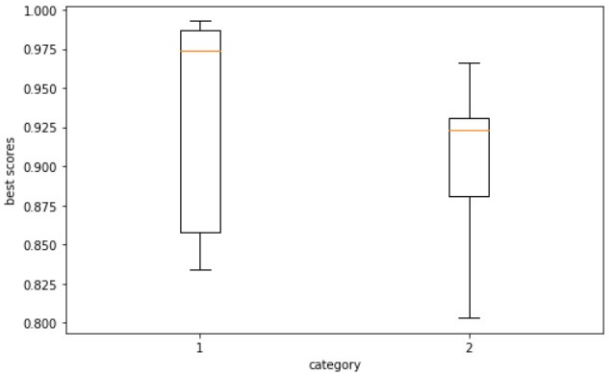

Fig. 5. Box plots of the best score by categories. Category 1 includes score functions that do not use uncertainty (S. and B.A.). Category 2 includes the score function using uncertainty (W.S. and W.B.A.). The LRS dataset was ignored given its very low scores.

TABLE II
EVALUATION OF THE DIFFERENT SCORE FUNCTION.

Dataset		AUC ROC for given score function			
Name	% Anomaly	S.	B.A.	W.S.	W.B.A.
LRS	2.2	0.7557	0.7571	0.7561	**0.7577**
	5.4	0.6959	0.6932	**0.6971**	0.6942
	10	0.6408	**0.6506**	0.6375	0.6458
Isolet	1.2	0.9075	0.909	0.914	**0.9148**
	3.5	0.8743	0.8718	**0.8814**	0.8807
	6.8	0.8115	0.8101	**0.8193**	0.8173
Satimage	2.9	0.9386	0.936	**0.966**	0.9645
	5.7	0.9159	0.9061	**0.9469**	0.9383
	10.8	0.824	0.805	**0.8607**	0.8447
Ionosphere	4.6	0.9927	**0.9930**	0.9924	0.9913
	10.6	**0.9743**	0.9725	0.9718	0.9706
	18.8	0.985	**0.9868**	0.9831	0.9837
WBC	5.2	0.9212	0.9188	0.9268	**0.9316**
	9.7	0.9184	0.918	0.9212	**0.9236**
	15	0.7748	0.7796	0.788	**0.8036**
Musk	7.2	0.9868	**0.9871**	0.9854	0.9854
	10.1	0.9288	0.9277	**0.931**	0.93
	16.3	0.9269	0.9263	0.9285	**0.9289**
Letter Recognition	1.9	**0.8887**	0.887	0.8884	0.8866
	3.3	0.8572	**0.8583**	0.847	0.8463
	6.3	0.8327	**0.834**	0.8247	0.8256

V. CONCLUSION

In this paper, we reviewed various function score applicable to neural networks in the context of anomaly detection. As shown in the part IV-C, the classic score function generally shows poorer results than others. This result encourages the use of the Bayesian approximation method in autoencode neural networks. Indeed, the other functions show rather similar performances. In addition, setting up an autoencoder that can produce a Bayesian approximation, as explained in Section II-B1, is the largest part of the work to use the proposed new score functions. Moreover, the complexity differs very little between the calculation of the Bayesian approximation and the calculation of the proposed functions. We therefore encourage the consideration of our functions as well as the Bayesian approximation when setting up an autoencoder used in the context of anomaly detection.

It would be interesting to study the performance of the score functions with different datasets, needing other types of autoencoders. Future works to be carried out would consist in repeating a similar experiment on the performances of functions applied, for example, to convolutional autoencoders or to recurrent autoencoders used for anomaly detection.

VI. ACKNOWLEDGEMENTS

The authors thank the company Zenika and especially the agency of Lille for funding this work. This work was done as part of a thesis carried out at the MIS laboratory of the University of Picardie Jule Vernes.

REFERENCES

[1] V. Kumar, "Parallel and distributed computing for cybersecurity," *IEEE Distributed Systems Online*, vol. 6, 2005.
[2] C. Spence, L. Parra, and P. Sajda, "Detection, synthesis and compression in mammographic image analysis with a hierarchical image probability model," *Proceedings IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA 2001)*, 2001.
[3] E. Aleskerov, B. Freisleben, and B. Rao, "Cardwatch: A neural network based database mining system for credit card fraud detection," *Proceedings of IEEE Computational Intelligence for Financial Engineering.*, 1997.
[4] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, 2014.
[5] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: Behavior-based malware detection system for android," *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*, 2011.
[6] P. García-Teodoroa, J. Díaz-Verdejoa, G. Maciá-Fernándeza, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security*, 2009.
[7] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.
[8] K. Thangavel and A. Pethalakshmi, "Dimensionality reduction based on rough set theory: A review," *Applied Soft Computing*, 2009.
[9] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, 1987.
[10] M. A. Carreira-Perpinán, "A review of dimension reduction techniques," 1997.
[11] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, 2006.
[12] J. Almotiri, K. Elleithy, and A. Elleithy, "Comparison of autoencoder and principal component analysis followed by neural network for e-learning using handwritten recognition," *IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, 2017.
[13] L. van der Maaten, E. Postma, and J. van den Herik, "Dimensionality reduction: A comparative review," *Journal of Machine Learning Research*, 2009.
[14] J. Li, M.-T. Luong, and D. Jurafsky, "A hierarchical neural autoencoder for paragraphs and documents," *arXiv preprint arXiv:1506.01057*, 2015.
[15] J. Masci, U. Meier, D. Ciresan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," *International Conference on Artificial Neural Networks (ICANN)*, 2011.
[16] M. Chen, X. Shi, Y. Zhang, D. Wu, and M. Guizani, "Deep feature learning for medical image analysis with convolutional autoencoder neural network," *IEEE Transactions on Big Data*, 2016.
[17] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," *International Conference on Machine Learning*, 2016.
[18] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length and helmholtz free energy," *Advances in Neural Information Processing Systems (NIPS)*, 1993.
[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016, ch. Autoencoders.
[20] V. J. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial Intelligence Review*, 2004.
[21] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection : A survey," *ACM Computing Surveys (CSUR)*, 2009.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, 2014.

[23] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," *International Conference on Machine Learning*, 2008.

[24] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," *Workshop on Machine Learning for Sensory Data Analysis (MLSDA)*, 2014.

[25] R. M. Neal, "Bayesian learning for neural networks." PhD dissertation, University of Toronto, 1995.

[26] D. J. MacKay, "A practical bayesian framework for backpropagation networks," *Neural Computation*, 1992.

[27] L. Zhu and N. Laptev, "Deep and Confident Prediction for Time Series at Uber," *arXiv preprint arXiv:1709.01907v1*, 2017.

[28] J. Paisley, D. Blei, and M. Jordan, "A practical bayesian framework for backpropagation networks," *Proceedings of the 29th International Conference on Machine Learning*, 2012.

[29] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *The International Conference on Learning Representations*, 2014.

[30] J. M. Hernández-Lobato and R. Adams, "Probabilistic backpropagation for scalable learning of bayesian neural networks," *Proceedings of the 32nd International Conference on Machine Learning*, 2014.

[31] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[32] M. Fortunato, C. Blundell, , and O. Vinyals, "Bayesian recurrent neural networks," *arXiv preprint arXiv:1704.02798*, 2017.

[33] J. M. Hernández-Lobato, Y. Li, M. Rowland, D. Hernández-Lobato, T. Bui, and R. E. Turner, "Black-box α-divergence minimization," *Proceedings of the 33rd International Conference on Machine Learning*, 2016.

[34] Y. Li and Y. Gal, "Dropout inference in bayesian neural networks with alpha-divergences," *arXiv preprint arXiv:1703.02914*, 2017.

[35] Y. Gal, "Uncertainty in deep learning," PhD dissertation, University of Cambridge, 2016.

[36] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," *Advances in Neural Information Processing Systems*, 2016.

[37] A. Zimek, M. Gaudet, R. J. G. B. Campello, and J. Sander, "Subsampling for efficient and effective unsupervised outlier detection ensembles," *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

[38] D. Chapman and A. Jain, "Musk (version 2) data set," 1995. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Musk+(Version+2)

[39] W. H. Wolberg, W. N. Street, and O. L. Mangasarian, "Breast cancer wisconsin (diagnostic) data set," 1995. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

[40] V. Sigillito, "Breast cancer wisconsin (diagnostic) data set," 1989. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/ionosphere

[41] J. Stutz, "Low resolution spectrometer data set," 1988. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Low+Resolution+Spectrometer

[42] R. Cole, M. Fanty, and T. Dietterich, "Isolet data set," 1988. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/isolet

[43] A. Srinivasan, "Statlog (landsat satellite) data set," 1993. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Statlog+\%28Landsat+Satellite\%29

[44] D. J. Slate, "Letter recognition data set," 1991. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/letter+recognition

[45] A. Frank and A. Asuncion, "Uci machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[46] J. A. Hanley and B. J. McNeil, "The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve," *Radiology*, 1982.