HAL open science

Detecting covariate shift with Black Box predictors

Florence Alberge, Clément Feutry, Pierre Duhamel, Pablo Piantanida

To cite this version:

Florence Alberge, Clément Feutry, Pierre Duhamel, Pablo Piantanida. Detecting covariate shift with Black Box predictors. International Conference on Telecommunications (ICT 2019), Apr 2019, Hanoi, Vietnam. 10.1109/ICT.2019.8798827 . hal-02172275v3

HAL Id: hal-02172275

https://centralesupelec.hal.science/hal-02172275v3

Submitted on 22 Jun 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Detecting Covariate Shift with Black Box Predictors

Florence Alberge*, Clément Feutry*, Pierre Duhamel* and Pablo Piantanida*†
*Laboratoire des Signaux et Systèmes (L2S), CentraleSupélec-CNRS-Université Paris-Sud, Université Paris-Saclay, Gif-sur-Yvette, France
†Montreal Institute for Learning Algorithms (MILA), Université de Montréal, QC, Canada
Email: {florence.alberge,clement.feutry}@l2s.centralesupelec.fr

***Abstract*—Many Machine Learning algorithms aiming at classifying signals/images X among a number of discrete labels Y involve training instances, from which the predictor $P_{Y|X}$ is extracted according to the data distribution $P_{X|Y}$. This predictor is later used to predict the appropriate label for other instances of X that are hence assumed to be drawn from the same distribution. This is a fundamental requirement for many real-world applications, therefore it is of great importance to monitor the reliability of the classification provided by the algorithm based on the learned distributions, when the test set statistics differ from the training set ones. This paper makes a step in that direction by proposing a Black Box Shift Detector of the data evolution (covariate shift). 'Black Box' here means that it does not require any knowledge of the predictor's architecture. Experiments demonstrate accurate detection on different high-dimensional datasets of natural images.**

***Index Terms*—Machine learning, deep learning, classification, statistical hypothesis testing, proven reliability.**

I. Introduction

Deep learning with large enough labeled datasets has been highly successful in several applications such as image recognition, speech recognition, recommendation, machine translation [1] and more recently communications [2]. These methods implicitly assume the data of interest follow the same distribution as the one underlying training sequences. However, this is a very strong assumption since in many real-world applications such as communication networks the statistics of the data evolves over time. In most cases we have very little or even no prior knowledge about how the test distribution may shift, e.g., anticipated changes in the traffic or topology distributions. It is thus necessary to build a decision rule to detect such changes in order to take appropriate decisions (i.e. launch a new training).

This work addresses the problem of detecting changes of the conditional distributions, (from $P_{X|Y}$, estimated from the training data and $Q_{X|Y} \neq P_{X|Y}$, estimated from the test data) over the features X. This is done under covariate shift assumption in which both training and test distributions share the same marginal distribution P_Y over the labels Y (or concepts to be learnt). To this end, we propose a simple yet effective method which is applicable to any trained deep neural soft-classifier $P_{Y|X}$ describing the probability of the label Y given the features X. This method does not require joint training nor to access the details of the classifier itself, such as the intermediate representations or parameters. The difficulty of the present scenario relies on the derivation of a binary classifier only from labeled trained data from $P_{X|Y}$ equipped with the predictor $P_{Y|X}$ without accessing to data from the test distribution $Q_{X|Y}$. Our approach relies mainly on the inference of the cumulative probability function of the likelihood predictor[1]:

$$F(r|P_X) \equiv \mathbb{E}_Y \left[\Pr\left(-\log P_{Y|X}(Y|X) \leq r\right)\right], \qquad (1)$$

which can be used to relate covariate shift from P_X to Q_X by detecting the shift between $F(r|P_X)$ and $F(r|Q_X)$ based on the model misspecification induced by the predictor $P_{Y|X}$. A chi-squared test [3] is then implemented to determine whether there is a significant deviation between the expected frequencies from the likelihood predictor during training and the frequencies observed from test data.

There are two main contributions in this paper: First, we provide an improved understanding of the relation between covariate shift and predictor misspecification. Second, we provide a systematic method for checking covariate shift with black boxes. We demonstrate the effectiveness of the proposed method using deep feed-forward neural networks, trained for image classification tasks on various well-known datasets including: CIFAR10 [4], MNIST [5], SVHN [6] and Fashion-MNIST [7] which are widely used in the field (see Appendix for their precise content).

A. Related work

A similar scenario has been investigated in [8], in which both training and test distributions share the same conditional distribution $P_{Y|X}$, while their marginal distributions, P_X and Q_X, are different. The focus was on correcting the distribution shift while here we concentrate on the detection of this shift. Major efforts have been dedicated to importance reweighing (see [9] and references therein). These methods have been developed for detecting either out-of-distribution or adversarial samples, or both. An evaluation of the confidence one can have in a classifier prediction has been studied in [10]. A method for detecting any abnormal sample, which

This project has received funding from the European Unions Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No 797805.

[1]The probability on X is determined by the distribution P_X, which is not a function of the labels.

is applicable to any pre-trained softmax neural classifier in presence of labeled test samples was recently introduced in [11]. This approach is based on the characterization via the class conditional Gaussian distributions of the features of the deep models under Gaussian discriminant analysis, which result in a confidence score based on the Mahalanobis distance. This method outperforms recent works in [12], [13] based on the confidence from the posterior distribution. However, improved performance in [11] are obtained by measuring and combining the final features but also the low level features in the neural network. This comes in contradiction with the black-box assumption considered in this paper. While [12], [13] are based on the sole maximum probability at the softmax output, we prove here that considering the complete softmax output is beneficial for the detection of covariate shift.

While most of the research efforts have been towards improving classifier performance and compensating for covariate shift, detecting the presence of distribution shift in testing from unlabeled samples has received far less attention.

The rest of this paper is organized as follows. Section II introduces some basic concepts on neural networks, then defines the problem. Section III presents the Pearson's chi-squared test which is used to detect the covariate shift. Simulation results are provided in Section IV while Section V provides a summary with concluding remarks.

II. Preliminaries and Problem formulation

A. Review of neural networks

The central problem in machine learning and deep learning is to learn useful representations of the input data from exposure to known examples of inputs and outputs. Deep learning involves several successive layers of representation, based on so-called "neurons". In a layer of representation, each neuron collects as inputs a number of outputs of the previous layer and performs a so-called "neuron pre-activation function" as follows : $\mathbf{a}(\mathbf{x}) = (W\mathbf{x} + \mathbf{b})$ for a vector input $\mathbf{x}$ where W is a weight matrix, $\mathbf{b}$ is the vector bias. The output of the neuron is $f\big(\mathbf{a}(\mathbf{x})\big) = f(W\mathbf{x}+\mathbf{b})$ where f is the activation function (preferably non-linear).

A deep neural network is obtained by stacking layers one on top of another. Once the architecture of the network is defined, a training is performed, based on a set of known input/output pairs, using forward and backward gradient computations. Under appropriate conditions and computations (softmax classifier), the resulting output can be considered as an estimate of the probability of the labels for the considered input. We shall not go further into technical details since the neural network will be considered here as a black-box (for further details the reader is referred to [14]).

In this paper, the proposed method for detecting mismatched test data requires only output probabilities induced by the softmax without any knowledge of the network architecture or neither observations of intermediate layers outputs. In some sense, we are thus studying a "black box" method that should be able to assess whether the data processed by the network has statistical properties similar to the training data or not. Obviously, this gives a good view of the reliability of the classification results provided by the network. Therefore, in the following, we will use the validation set, e.g., a subset of the training set which was not used for training, as a reference and the test set as the database of matched data.

B. Problem formulation

Let us consider a pre-trained deep neural network with a softmax classifier denoted by $P_{Y|X}$. Let $x \in \mathcal{X}$ denote a feature input and let $y \in \mathcal{Y} \equiv \{0, 1, \ldots, C-1\}$ denote the corresponding output or label. The softmax classifier provides as an output the probability $P_{Y|X}(c|x)$ for each class label c given the feature x. Without loss of generality, the network is assumed to perform an image classification task. Therefore, the image can be interpreted as being a redundant representation of the class and the image classification problem could be seen as a data compression process. A binary instantaneous code of minimum expected-length could be constructed to describe the source with optimal coding length: $-\log_2 P_{Y|X}(c|x)$. Intuition behind this work is that the statistical distribution of the code lengths should reflect to some extend the statistical distribution of the inputs. In the following, the empirical cumulative distribution function $\hat{F}_n(r|\texttt{Data})$ of $-\log_2 P_{Y|X}$:

$$\hat{F}_n(r|\texttt{Data}) \equiv \mathbb{E}_Y\left[\frac{1}{n}\sum_{i=1}^{n}\mathbb{I}_{[0,r]}(Y|x_i)\right], \tag{2}$$

where $\mathbb{I}_{[0,r]}(y|x_i) = 1$ if $-\log_2 P_{Y|X}(y|x_i) \leq r$ and zero otherwise, is used in order to detect data shift. Obviously, assuming that the data samples come from distribution P_X, by the Glivenko-Cantelli theorem, it follows that

$$\sup_r \left|\hat{F}_n(r|\texttt{Data}) - F(r|P_X)\right| \longrightarrow 0$$

almost surely in the limit when n goes to infinity. Kolmogorov strengthened this result, by effectively providing the rate of this convergence. However, in practice, the statistic requires a relatively large number of data points (in comparison to other goodness of fit criteria) to properly reject the null hypothesis. This will be made clear in the following example.

Assume that detection is based on batches of images, which is indeed the case for many practical situations. Consider the empirical cumulative distribution function plotted in Fig. 1 for the case of CIFAR10 based on the neural network specified in Algorithm 3 relegated to Section VI. Two different variants are studied: $\hat{F}_n(r|\texttt{Matched test})$ which is the restriction of the predictor's likelihood to feature inputs x belonging to the test set and $\hat{F}_n(r|\texttt{Mismatched data})$, which is obtained by adding salt and pepper noise to `Matched test` with corruption probability 0.08. We use 10^4 input images and the CIFAR10 dataset consists of 10 classes. The size of each set is thus equal to 10^5. It may be observed in Fig. 1 that the range is almost the same for both sets. Whereas an hypothesis test based on the comparison of the value of $-\log_2 P_{Y|X}$ to a threshold would not be relevant in this case. The reason behind this fact relies on the property of the softmax classifier which is known to produce overconfident posteriors, even for

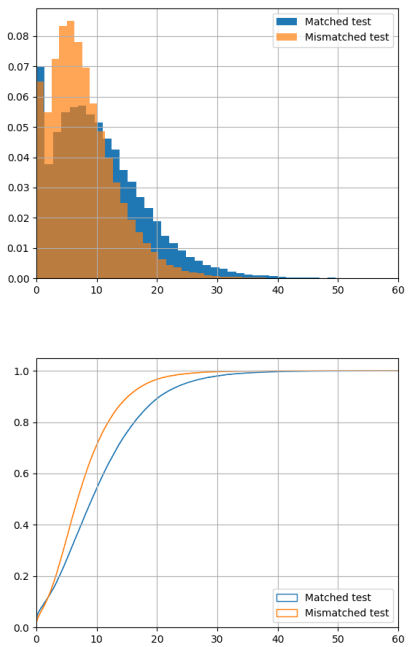

Fig. 1. Empirical distribution function (up) and empirical cumulative distribution function (down) $\hat{F}_n(r|\texttt{Matched test})$ and $\hat{F}_n(r|\texttt{Mismatched test})$, where `Matched test`$\equiv$ CIFAR10 and `Mismatched test`$\equiv$ CIFAR10+S&P.

misclassified samples [15]. In contrast, a test of homogeneity seems to be more appropriate here since the distributions exhibit different shapes. This test will be introduced in the next section.

III. Pearson's Chi-squared test

The method proposed here to detect a possible evolution of the data set is based on a chi-Square test for homogeneity [3]. This test compares the distribution of counts for groups using the same categorical variable, i.e., $-\log_2 P_{Y|X}$ is used here. The test determines whether frequency counts are distributed identically across different populations. Two groups are considered here. The first group, called `Standard` serves as a proper benchmark. `Standard` contains realizations obtained by sampling jointly the likelihood predictor function: $-\log_2 P_{Y|X}$, according to the label distribution P_Y and the feature samples X from input images, e.g., images from a validation set. The second group called `Sample` contains CN^{SMP} realizations of $-\log_2 P_{Y|X}$, where C is the number of classes and N^{SMP} is the number of image on this sample test. Obviously, we want an efficient test for small values of N^{SMP}. However, such minimal value will heavily depend on the statistical difference between the underlying distributions. The test procedure is described below.

A. Testing homogeneity

We return to the simple goodness of fit problem for categorical data that was briefly considered. Suppose that each sample $-\log_2 P_{Y|X}$ is divided into $|K|$ categories (e.g. defining $|K|$ real valued intervals). We would like to evaluate whether the distribution of categories in each group is the same. Let us denote the empirical estimates of the probabilities as $\Pr(K = k|\texttt{Standard}) \approx \hat{p}_k^{\text{STD}}$ and $\Pr(K = k|\texttt{Sample}) \approx \hat{p}_k^{\text{SMP}}$ and the corresponding probabilities in the population from which the sample groups are selected as p_k^{STD} and p_k^{SMP}. Then we want to test the following hypotheses:

$$\begin{cases} \text{H0} & : \quad p_k^{\text{STD}} = p_k^{\text{SMP}} = p_k, \\ \text{H1} & : \quad \text{otherwise.} \end{cases} \tag{3}$$

A standard test, proposed by Pearson (1900), rejects H0 for large values of Pearsons Chi-squared statistic:

$$T \equiv \sum_{k=1}^{|K|} \frac{(N_{k,th}^{\text{STD}} - N_k^{\text{STD}})^2}{N_{k,th}^{\text{STD}}} + \frac{(N_{k,th}^{\text{SMP}} - N_k^{\text{SMP}})^2}{N_{k,th}^{\text{SMP}}}, \tag{4}$$

where N_k^{STD} and N_k^{SMP} stand for the number of instances in the k-th category within each sample group, i.e., the evidence $N_k^{\text{STD}} = \hat{p}_k^{\text{STD}} N^{\text{STD}}$ and $N_k^{\text{SMP}} = \hat{p}_k^{\text{SMP}} N^{\text{SMP}}$. The theoretical numbers of instances are computed as follows: $N_{k,th}^{\text{STD}} \equiv p_k N^{\text{STD}}$ and $N_{k,th}^{\text{SMP}} \equiv p_k N^{\text{SMP}}$. Since the true probabilities $\{p_k\}$ are unknown, these are estimated as: $p_k \approx \frac{N_k^{\text{STD}} + N_k^{\text{SMP}}}{C(N^{\text{STD}} + N^{\text{SMP}})}$ which is the best estimator based on the observations and under the assumption that H0 holds. The total number of categories is chosen to be 10, with intervals of equal length. If either $N_{k,th}^{\text{STD}} < 5$ or $N_{k,th}^{\text{SMP}} < 5$ then the k-th category is grouped with an adjacent category and thus $|K| \leq 10$.

B. Covariate shift detection

As previously mentioned, `Standard` serves as a benchmark and characterizes the behavior of the pre-trained network with respect to the likelihood's predictor when input features and random labels are used. Therefore, a simple way to construct `Standard` is to aggregate the softmax outputs obtained with the validation set. `Sample` is obtained by randomly sampling the likelihood's predictor either from `Matched test` feature inputs, i.e., H0 holds, or from `Mismatched test` feature inputs, i.e., H1 holds and H0 should be rejected. For that purpose, T should be compared to the critical value. The critical value of the test is chosen according to the desired level of confidence and can be estimated from expression (2) according to the `Standard` feature inputs. The efficiency of this test is usually measured through the two error probabilities and for several critical values:

$$\begin{cases} \alpha(T) & \equiv \quad \Pr\left(\text{reject H0} \,|\, \text{SMP} \equiv \texttt{Matched test}\right), \\ \beta(T) & \equiv \quad \Pr\left(\text{accept H0} \,|\, \text{SMP} \equiv \texttt{Mismatched test}\right). \end{cases}$$

IV. Simulation results

A. Noisy data detection

In this section, the mismatched test data is modeled by an additive noise on the proper data. Namely, `Mismatched test` $= \{x + n, x \in$ `Matched test`$\}$

where n is a background noise with uniform distribution between 0 and $U_{max} = 30$ [16]. In the following, we use two datasets: MNIST, Fashion-MNIST. MNIST is a dataset of handwritten digits with 10000 testing examples and 60000 training examples (10% of these training examples from the validation set). Fashion-MNIST is a dataset of Zalando's article images divided in 10 classes with 60000 training examples (10% for validation set) and 10000 training set. Numerical results are given in Table I and II. The true negative rate (TNR) stands for the proportion of mismatched sequences that are properly identified, true positive rate (TPR) stands for proportion of matched sequences that are properly identified. AUROC stands for Area under Receiver Operating Characteristic and is computed from the plots Figure 2 and 3. The main trend in the results follows the intuition: the larger N^{SMP} is, i.e. the longer sequences are, the better the performances. The detection accuracy is over 95% with $N^{\text{SMP}} = 40$ on MNIST database. The results are even better with Fashion-MNIST database, the main reason for the differences is the nature of the network being different. The network used for MNIST has a simple feed-forward architecture with one layer and dropout. The accuracy of the classification task is 98.02% on the test set (matched sequence) and 96.75% on the noisy version of the test set (mismatched sequence) The Fashion-MNIST network involved a deeper and more complex architecture based on convolutional layers with dropout, details are in the appendix (see section VI).The accuracy on the classification task is 91.40% on the matched sequence (test set) and 90.30% on the mismatched sequence (noisy test set). The 95% detection accuracy (detecting true matched sequence and true mismatched sequence) is reached with only $N^{\text{SMP}} = 20$ on this database. The method is efficient to separate target data and mismatched data even when the difference between them is a rather small disturbance. These results prove the efficiency of our method, but also demonstrate that it can be applied to any neural network outputs regardless of their architecture complexity, truly behaving as a black box covariate shift detector.

N^{SMP}	TNR at TPR 95%	AUROC	Detection accuracy
10	48.82	83.93	80.24
20	75.96	94.40	89.05
30	90.88	97.82	93.80
40	95.76	99.04	95.77
50	98.6	99.62	97.81
60	99.58	99.88	98.79
70	99.88	99.97	99.34

TABLE I
RESULTS ON THE MNIST DATASET FOR SEVERAL VALUES OF N^{SMP} WHEN MISMATCHED IS ADDITIVE BACKGROUND NOISE

Plots of figure 2 and 3 show the accuracy trade-off for different values of N^{SMP} respectively for MNIST and Fashion-MNIST. The performances shown here are quite good even with a small sequence ($N^{\text{SMP}} \leq 20$). For the higher value of N^{SMP} the detection of mismatched data can be made with high confidence (the probability is over 0.9995). It seems that for a wide range of application one can still find the proper N^{SMP} value to satisfy its needs.

N^{SMP}	TNR at TPR 95%	AUROC	Detection accuracy
10	52.76	88.74	84.87
20	92.72	98.40	95.09
30	97.62	99.40	96.87
40	99.6	99.80	98.45
50	99.78	99.93	99.01
60	99.88	99.95	99.20
70	> 99.96	99.996	99.71

TABLE II
RESULTS ON THE FASHION-MNIST DATASET FOR SEVERAL VALUES OF N^{SMP} WHEN MISMATCHED DATA IS ADDITIVE BACKGROUND NOISE

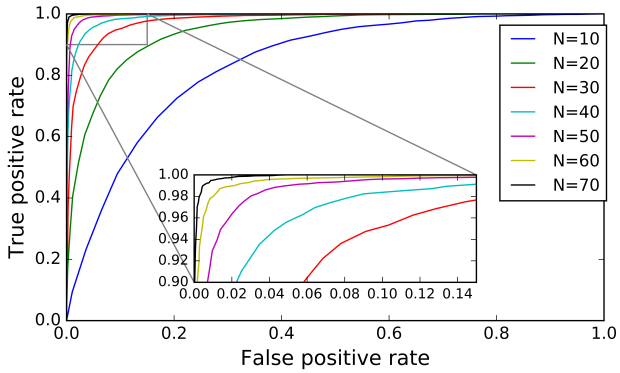

Fig. 2. Accuracy trade-off for different length of sequence N^{SMP} on MNIST: true positive rate (correctly identifying normal samples) as a function of false positive rate (missing the detection of abnormal samples).

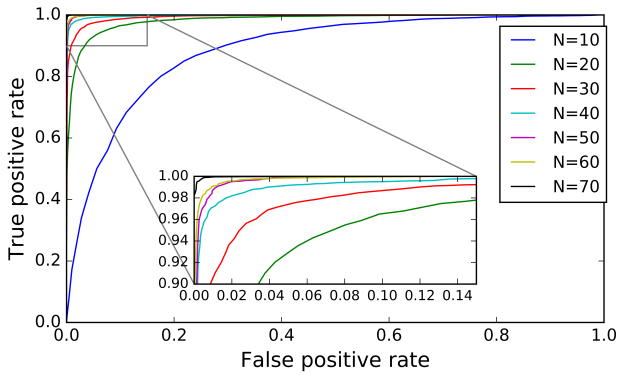

Fig. 3. Accuracy trade-off for different length of sequence N^{SMP} on Fashion-MNIST: true positive rate (correctly identifying normal samples) as a function of false positive rate (missing the detection of abnormal samples).

B. Out-of-distribution samples detection

We turn now to the detection of out-of-distribution samples. These abnormal samples can be due to an adversarial attack or to an error of manipulation. In this context, the images in `Mismatched test` and in `Matched test` are significantly different therefore the test of homogeneity is expected

to be efficient for smaller values of N^{SMP} than in the previous context. The dataset we are using in this section are CIFAR10 and SVHN. CIFAR-10 contains 32x32 colour images belonging to 10 different classes classes characterizing the picture (train, horse, plane,...), with 50000 training examples (10% for validation set) and 10000 testing examples. The Street View House Numbers (SVHN) dataset is obtained from house numbers in Google Street View images. Each example is a 32x32 colored image, associated with a label from 10 classes as in CIFAR-10. The network used are a deep convolutional network (detailed in section VI) trained on sample from one database and the mismatched test are perform with sample from the other database. The classification accuracies are 73.60% and 91.64% on respectively the CIFAR10 dataset and the SVHN dataset.

N^{SMP}	TNR at TPR 95%	AUROC	Detection accuracy
5	86.22	95.68	91.34
7	95.88	98.48	95.88
9	98.28	99.39	97.71
11	99.34	99.74	98.47
13	99.64	99.80	99.52
15	99.98	99.99	99.81
17	>99.99	>99.99	99.87

TABLE III
RESULTS ON A SVHN TRAINED NETWORK TESTED ON CIFAR10 SAMPLES FOR SEVERAL VALUES OF N^{SMP}

N^{SMP}	TNR at TPR 95%	AUROC	Detection accuracy
5	27.14	86.69	82.47
7	55.26	92.96	88.19
9	84.8	96.80	92.72
11	88.28	97.49	93.75
13	91.26	97.84	94.62
15	98.58	99.19	96.96
17	98.72	99.28	97.06
19	99.44	99.45	97.55

TABLE IV
RESULTS ON A CIFAR10 TRAINED NETWORK TESTED ON SVHN SAMPLES FOR SEVERAL VALUES OF N^{SMP}

The error probabilities are given in Tables III and IV. In all studied cases, the method proves its efficiency. Indeed, with 7 or 15 images (respectively when CIFAR10 and SVHN samples are the mismatched data) it is possible to take apart normal from abnormal samples with an accuracy over 95%. One can observe that the performance is quite different in both situations of a network trained on SVHN, tested on CIFAR10, and the converse situation. This has a simple explanation : If a network has poor performance on the training set, it will be difficult to distinguish legitimate signals from illegitimate ones. And this is the case here : the network trained on SVHN has a performance of 91 %, while this performance decreases to 73 % when the network is trained on CIFAR10. The curves on the figures 4 and 5 look less smooth because N^{SMP} is small compared to the value of used for the figure 2 and 3. They may overlap because they are plotted with only a step of two between values of N^{SMP}. Note that the hypothesis are only valid if $N^{\text{SMP}} \geq 5$. As shown on figure 4 the case $N^{\text{SMP}} = 5$ can produce surprising shaped curve but still bring relevant information. The method is efficient to track out-of-distribution sample sequences quickly and with a strong confidence even if the network does not perform perfectly on the regular task: The Cifar10 trained network output statistics, despite not coming from top of the art classifier, still enable an accurate detection with as few as $N^{\text{SMP}} = 15$. Finally as shown on the last two lines of table III top performances are achieved for quite small value of N^{SMP}: the difference between $N^{\text{SMP}} = 15$ $N^{\text{SMP}} = 17$ (reinforced by their respective curves in figure 4) is faint which makes sense because the performances are close to 100%

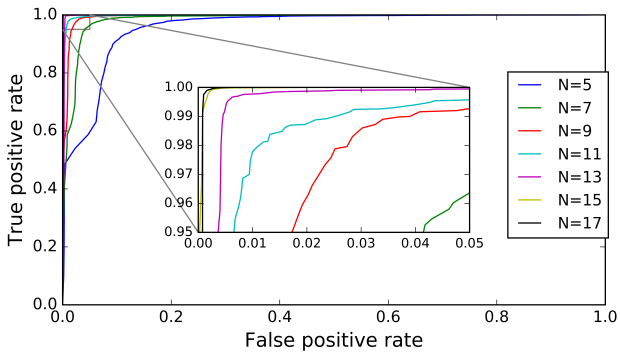

Fig. 4. Accuracy trade-off for different length of sequence N on a network trained on SVHN and tested with CIFAR10 images: true positive rate (correctly identifying normal samples) as a function of false positive rate (missing the detection of abnormal samples).

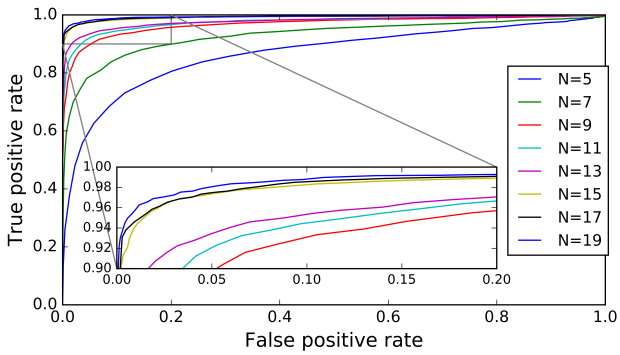

Fig. 5. Accuracy trade-off for different length of sequence N on a network trained on CIFAR10 and tested with SVHN images: true positive rate (correctly identifying normal samples) as a function of false positive rate (missing the detection of abnormal samples).

V. CONCLUSION

A test of homogeneity has been proposed for the detection of either data evolution or of abnormal samples in a classification context. The method is based on the assumption that the output distribution of the neural network's softmax classifier reflects the statistical properties of the input, and can

serve as a basis for evaluating some "distance" between the statistical properties of the signals/images under test compared to the reference situation, given by the training sequences. This approach has been validated through simulations, demonstrating that good homogeneity checks can be obtained even for small number of observations. The method is very simple and is applied independently of the intrinsic neural network architecture. In fact, the neural network is considered as a black-box since the method necessitates the sole knowledge of the outputs. This comes in contrast with most solutions in the literature.

VI. Appendix

The Neural Network Architectures used in the paper are given below. Dense(n) denotes a fully-connected layer with n output units. Con2D(n, $w \times h$) denotes a convolutional layer with n output features and filter size of $w \times h$. ReLU is the rectified linear unit activation. The implementation is based on Keras.

Algorithm 1 MNIST

```
model = models.Sequential()
model.add(Dense(512),activation='relu')
model.add(Dropout(0.2))
model.add(Dense(10))
model.add(Activation('softmax'))
```

Algorithm 2 Fashion-MNIST

```
model = models.Sequential()
model.add(Conv2D(64, (2, 2), activation='relu',
input_shape=(28, 28, 1)))
model.add(MaxPool2D())
model.add(Dropout(0.3))
model.add(Conv2D(32, (2, 2), activation='relu'))
model.add(MaxPool2D())
model.add(Dropout(0.3))
model.add(Flatten())
model.add(Dense(256),activation='relu')
model.add(Dropout(0.5))
model.add(Dense(10))
model.add(Activation('softmax'))
```

Algorithm 3 CIFAR-10 or SVHN

```
model = models.Sequential()
model.add(Conv2D(32, (3, 3), activation='relu',
input_shape=(32, 32, 3)))
model.add(MaxPool2D())
model.add(Dropout(0.2))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPool2D())
model.add(Dropout(0.2))
model.add(Flatten())
model.add(Dense(512,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(10))
model.add(Dropout(0.2))
model.add(Activation('softmax'))
```

References

[1] Y. LeCun, Y. Bengio, and G. E. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[2] H. Kim, Y. Jiang, R. B. Rana, S. Kannan, S. Oh, and P. Viswanath, "Communication algorithms via deep learning," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=ryazCMbR-

[3] E. L. Lehmann and J. P. Romano, *Testing statistical hypotheses*, 3rd ed., ser. Springer Texts in Statistics. New York: Springer, 2005.

[4] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[5] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[6] Y. Netzer, T. Wang, A. Coates, R. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning."

[7] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017. [Online]. Available: http://arxiv.org/abs/1708.07747

[8] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of Statistical Planning and Inference*, vol. 90, no. 2, pp. 227–244, Oct. 2000. [Online]. Available: http://www.sciencedirect.com/science/article/B6V0M-4136355-5/1/6432c256e0be03b1503bbf79e4e91d1a

[9] M. Sugiyama and M. Kawanabe, *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation*. The MIT Press, 2012.

[10] H. Jiang, B. Kim, M. Y. Guan, and M. R. Gupta, "To trust or not to trust A classifier," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, 2018, pp. 5546–5557. [Online]. Available: http://papers.nips.cc/paper/7798-to-trust-or-not-to-trust-a-classifier

[11] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *NeurIPS*, 2018, pp. 7167–7177.

[12] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *ICLR*, 2017.

[13] L. Y. Liang, Shiyu and R. Srikant, "Principled detection of out-of-distribution examples in neural networks," in *Sixth International Conference on Learning Representations*, 2018.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, book in preparation for MIT Press. [Online]. Available: http://www.deeplearningbook.org

[15] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Thirty-second Conference on Neural Information Processing Systems*, Montreal, Canada, 2018.

[16] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.