A Simple Statistical Method to Detect Covariate Shift

Clément FEUTRY[1], Pablo PIANTANIDA[1,2], Florence ALBERGE[1], Pierre DUHAMEL[1]

[1]Laboratoire des Signaux et Systèmes (L2S), CentraleSupélec-CNRS-Université Paris-Sud, Université Paris-Saclay,
Gif-sur-Yvette Cedex, France

[2]Montreal Institute for Learning Algorithms (Mila), Université de Montréal, QC, Canada
clement.feutry@l2s.centralesupelec.fr

Résumé – L'utilisation croissante de réseaux de neurones dans le traitement automatique de données soulève de nouvelles problématiques. L'une d'elles est la détection de décalage des covariables d'entrées. Ce décalage entraîne une inadéquation entre le réseau et les données: le réseau continue de traiter les données, mais fournit des résultats vides de sens. Nous présentons des méthodes statistiques de détection de comportement peu confiant.

Abstract – The growing use of neural networks for automatic data processing raises new challenges. Among this is detecting changes of distribution or covariate shift of the inputs. The shift leads to a mismatch between the network and the incoming data: the network makes decisions but its outputs are meaningless. This paper introduces a novel statistical method to detect under-confident behavior of the network.

1 Introduction

An arguable flow of the widely used neural network classifier is the fact that they always outputs an answer. It means that whatever the input, the network will process it and will assign a class. Putting a dog picture in a written digit classifier will results in the network outputting a digit, the most likely digit, even if it is a complete nonsense. This paper aims at identifying these phenomenons and therefore prevent them. Our main contribution is the usage of simple statistical tools to assess covariate shift without any other assumption on the network that it output a probability vector for each input. An empirically study shows that one of these tools performs better than the other ones when testing systematically the method on black boxes. We demonstrate the effectiveness of the proposed method using deep feed-forward neural networks, trained for image classification tasks on well-known datasets : CIFAR10 [1] and SVHN [2].

1.1 Related work

A similar scenario has been investigated in [3], 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 the shift itself. Major efforts have been dedicated to importance reweighing (see [4] and references therein). These methods have been developed for detecting either out-of-distribution or adversarial samples, or both. An evaluation of the confidence that can be obtained in a classifier task has been studied in [5]. A method for detecting abnormal samples, which is applicable to any pre-trained softmax neural classifier in presence of labeled test samples, was recently introduced in [6]. 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 results in a confidence score based on the so-called Mahalanobis distance. The method outperforms recent works in [7, 8] based on the confidence from the posterior distribution. However, improved performance in [6] 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 [7, 8] 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 the statistical tools, then defines the problem. Section III presents the simulations results while Section IV provides a summary with concluding remarks.

Notations

Dataset is noted by $\mathcal{X}$ and label alphabet is noted as $\mathcal{Y} \equiv \{0, 1, \ldots, C-1\}$. $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ is a sample and its associated label. The following probabilities are defined as follows : P_X, Q_X and P_Y are respectively the distribution over the data samples, the distribution of abnormal samples and the distribution of the labels.

2 Statistical Analysis

2.1 Problem formulation

We focus on the scenario of black box in which we do not have access to the network but only on its output soft probability, i.e., a probability distribution over the labels : $P_{\hat{Y}|X}$. A change in the distribution of X in this case from P_X to Q_X should result in a change of the conditional distribution outputted by the network from $P_{\hat{Y}|X}$ to $Q_{\hat{Y}|X}$. We suppose that we know the empirical distribution of P_{XY} from the dataset (training set and validation set), but we do not have any information about the distribution of Q_X (neither the true nor the empirical distribution). The main idea is to develop a method that can provide a meaningful statistic for the decision rule. To this end, the samples are grouped into batches of size N. The decision rule should use only the outputs of the neural network : $P_{\hat{Y}|X}(\hat{y}|x_i)$. The test statistics (to be defined in the next section) is computed based on the values of $z_i = P_{\hat{Y}|X}(\hat{y}|x_i)$ for $x_i, i \in [0, N-1]$ for the considered batch. Accepting the null hypothesis (H_0) means that the batch is considered as in-distribution (P_X). The type I error means that the hypothesis H_0 is rejected whereas it was true while the error of type II implies that the hypothesis H_1 is rejected while it was true.

2.2 Towards useful statistics

The key idea is that we are not interested in the network classification rate, we only assume that the network is fairly trained on dataset's classification task. What we are interested in is the confidence of the predictions expressed by the network on the data we feed it with. If the trained network computes the estimated label distribution on a mismatched sample, the network should show sign of a unusual uncertainty. Conversely, if the network processes an adapted sample, even if there might have some hesitations between several possible values of $\hat{Y}$, the network will be confident enough to dismiss some other $\hat{Y}$ values. This would imply that the previous statistics will asymptotically diverge. This raises three questions : how to measure this statistic divergence, how far apart will they diverge and how large should N (the number of test samples) be to assess the asymptotic effect? The processing of the data in batches of size N leads to a significant amount of values for each batch, each output being already of size C. The tools we will used are means measuring the level of confidence of a classifier computed over all the N samples of a batch :

- The quadratic mean : $\text{QM} = \sqrt{\dfrac{\sum_i^n z_i^2}{n}}$, ‘
- the arithmetic mean : $\text{AM} = \dfrac{\sum_i^n z_i}{n}$
- the geometric mean : $\text{GM} = \sqrt[n]{\prod_i^n z_i}$,
- the harmonic mean : $\text{HM} = \dfrac{n}{\sum_i^n \frac{1}{z_i}}$.

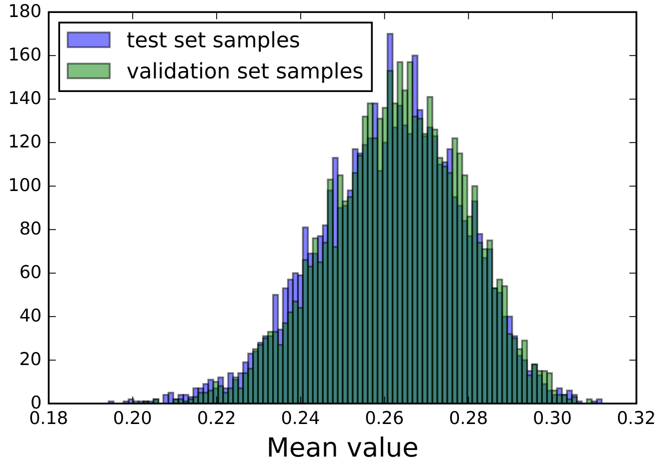

FIGURE 1 – Histogram comparing the predictions distributions ($P_{\hat{Y}|X}$) between test set and validation set for the quadratic mean.

Provided that all (x_i) are positive, these statistics satisfy :

$$\text{QM} \geq \text{AM} \geq \text{GM} \geq \text{HM}.$$

Since $\sum_{i=1}^{N} z_i = 1$ (normalized probability), the arithmetic mean is useless.

2.3 Statistical behaviour

The distribution of the training set, the validation set and the test set should be the same. Therefore the distributions of the network output for each of these sets should be the same or extremely close as can be seen on Fig. 1. This histogram shows that there is no distribution difference in the predictions for batches from validation set and test set : indeed the two distributions are hard to distinguish. Therefore the validation set distribution is a good approximation of the test set distribution. We also used matched and mismatched samples to confront the behavior of the different means in the histograms plotted in Fig. 2. The plot shows that the statistics have different distributions. As expected the values taken by the quadratic mean are higher than the other ones. Even if these statistics looks more distinguishable in the histogram, a scaling effect may induce a wrong intuition regarding lower values statistics. To properly rank the statistics, the results are explicitly summarized in Table 1 for several values of N. In order to rank this method we used the optimal threshold, i.e., we used here the knowledge of mismatched data statistics. The optimal threshold, knowing both P_X and Q_X is the threshold that result in the equal probability for type I and type II errors. The QM outperforms better the other averaging methods in this particular set-up.

3 Experiments

In this section, the distribution of Q_X is assumed unknown. Moreover we consider having access only to the training and validation data, meaning we have partial information over the

TABLE 1 – Performances of different means as a statistic. Percentage of error of both types over matched and mismatched data. The mismatched data statistic is known in order to find the optimal threshold. Maximum value is indicated in bold.

n	2	3	4	5	6
QM (%)	**18.07**	12.69	**9.57**	**6.51**	**5.47**
GM (%)	19.53	**12.59**	9.85	6.55	5.67
HM (%)	27.85	19.89	17.55	15.36	16.04

n	7	8	9	10
QM (%)	**3.77**	**3.35**	**3.19**	2.39
GM (%)	4.17	3.61	3.29	**2.01**
HM (%)	13.07	12.79	12.72	8.69

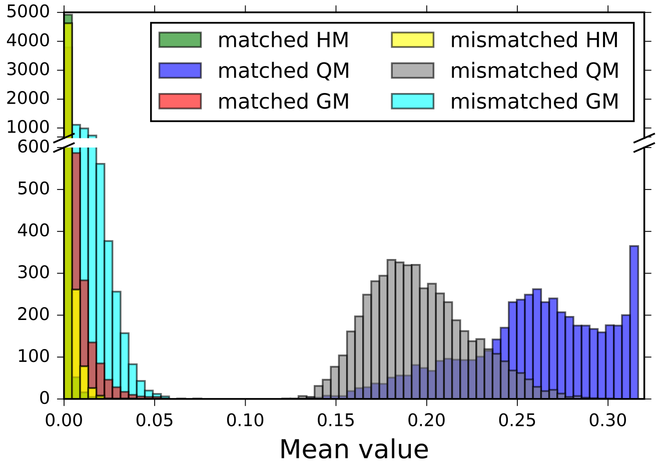

FIGURE 2 – Histogram exposing the behavior of different computed means on matched and mismatched data with $N = 2$. The least overlapping pair of histogram is the pair corresponding to the QM as shown on table 1.

distribution of P_X. Indeed we just know the empirical distribution of some of the x_i and we will only use the statistic of the validation data. Given the conditional distribution of the validation data $P_{\hat{Y}|X}(\hat{y}|x_i)$ and using a chosen value for the false discovery rate we can compute a threshold value for the hypothesis testing : for any batch, the relative value of its statistic is compared to the threshold in order to make a decision between H_0 and H_1.

TABLE 2 – Mean method and N value induced variation of the false discovery rate (FDR) for test set with a threshold computed on the validation set for a SVHN trained network.

FDR	5%		0.5%	
	min	max	min	max
QM (%)	2.22	3.68	0.04	0.16
GM (%)	1.42	3.00	0.02	0.1
HM (%)	1.52	2.88	0.0	0.16

3.1 SVHN database

The network is trained on the SVHN database. The mismatched data correspond to images from the CIFAR10 database. Therefore the two sets of data (matched and mismatched) are different in nature : the classifier could not produce any meaningful results with mismatched data. The values presented below are obtained with the network detailed in the appendix 4. Values are averaged over ten independent trainings of the network. Fig. 3 shows the influence of both the mean method and the N value. As expected, the higher N is the more likely mismatched data will be correctly filtered out. The quadratic mean (red curves) exhibits the best results, and the harmonic mean (black curves) performs quite worse comparatively. The geometric mean performance is mostly in between. The exposed method performs well, for example, with an FDR equal to 5%, 69.7% of batches are rightly filtered out with as little as 2 images batches, at the cost of 3.48% images of the test batches.

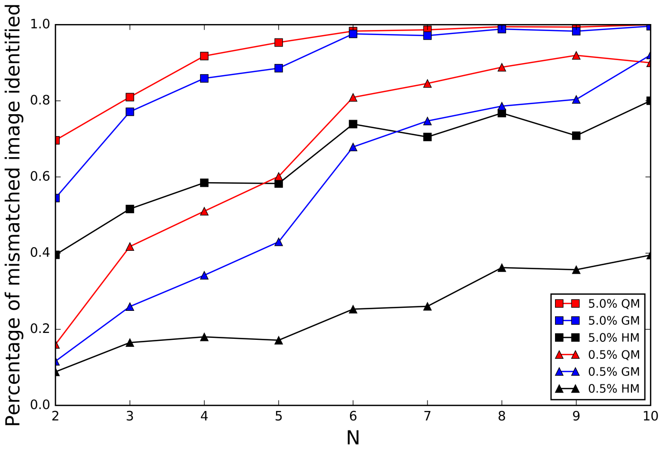

FIGURE 3 – Percentage of mismatched batches properly identified as a function of the size of batches for a SVHN trained network. The squares and triangles curves correspond to a false discovery rate of respectively 5% and 0.5%. The black, blue and red curves are respectively associated with the harmonic mean, the geometric mean and the quadratic mean. Values of the fixed false discovery rate are obtained using validation, the corresponding false discovery rate on the test sample is given on the table 2 .

3.2 CIFRAR10 database

Conversely of the previous experiment, the network is trained on the CIFAR10 database and the mismatched data correspond to images from the SVHN database. Here again the classifier could not produce any meaningful results with mismatched data. Values presented below are obtained with the network detailed in the Appendix. Values are averaged over ten trainings of the network. Fig. 4 shows the influence of both the mean method and the N value. The same behaviour is exposed

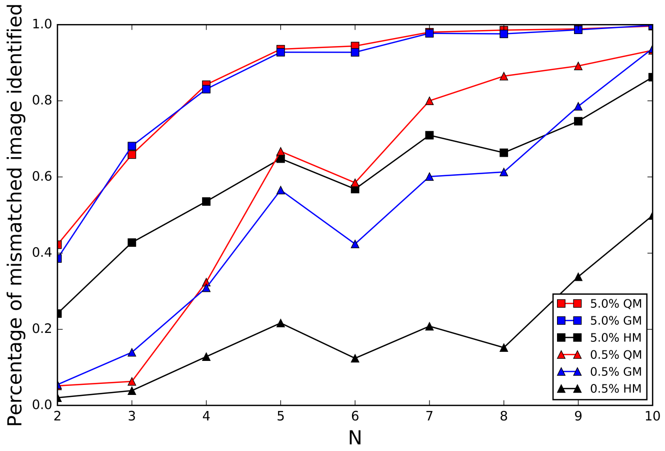

FIGURE 4 – Percentage of mismatched batches properly identified as a function of the size of the batch for a CIFAR10 trained network. The squares and triangles curves correspond to a false discovery rate of respectively 5% and 0.5%. The black, blue and red curves are respectively associated with the harmonic mean, the geometric mean and the quadratic mean. Values of the fixed false discovery rate are obtained using validation sample, the corresponding false discovery rate on the test sample is given on the table 3 .

here : the quadratic mean (red cuvres) shows the best results, and the harmonic mean (black curves) performs quite worse comparatively. The geometric mean performance is mostly in between. The exposed method performs a little worse than in the previous experiment : with an FDR of 5%, only 42% of batches are rightly filtered out with 2 images batches, at the cost of 5.64% images of the test batches. Table 3 and 2 shows different results : for the network trained on SVHN, FDR on the test set were always under fixed validation set FDR. On CIFAR10 values of FDR on the test set are around fixed validation set FDR, but not strictly under.

TABLE 3 – Mean method and N value induced variation of the false discovery rate (FDR) for test set with a threshold computed on the validation set for a CIFAR10 trained network.

FDR	5%		0.5%	
	min	max	min	max
QM (%)	5.1	6.58	0.28	1.3
GM (%)	4.48	6.84	0.26	1.02
HM (%)	4.06	6.4	0.12	0.7

4 Conclusion

This studies shows that the use of the quadratic mean is quite efficient to detect a shift in the distribution between trained data and unknown test data. This tool could be proven really useful because it manages to assess the confidence in a robust manner regarding its performances. This tool could be used to monitor shift in the tested data stream and indicate the right moment to perform a partial or total retraining of the network.

Appendix : Architectures

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 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'))
```

Références

[1] A. Krizhevsky, “Learning multiple layers of features from tiny images,” 2009.

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

[3] 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.

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

[5] H. Jiang, B. Kim, M. Y. Guan, and M. R. Gupta, “To trust or not to trust A classifier,” in *NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, 2018, pp. 5546–5557.

[6] 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.

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

[8] L. Y. Liang, Shiyu and R. Srikant, “Principled detection of out-of-distribution examples in neural networks,” in *ICRL*, 2018.