BRIOxAlkemy: A Bias detecting tool*

Greta Coraglia[1], Fabio Aurelio D'Asaro[3], Francesco A. Genco[1], Davide Giannuzzi[2], Davide Posillipo[2], Giuseppe Primiero[1] and Christian Quaggio[2]

[1]*LUCI Group, University of Milan, via Festa del Perdono 7, 20122 Milan, Italy*
[2]*Alkemy, …*
[3]*Verona*

Abstract

We present a tool for the detection of biased behaviours in AI systems. Using a specific probability-based algorithm, we provide the means to compare the action of the user's algorithm of choice on a specific feature that they deem "sensible" with respect to fixed classes and to a known optimal behaviour.

Keywords

LaTeX class, paper template, paper formatting, CEUR-WS

1. Introduction

The project BRIO aims at developing formal and conceptual frameworks for the analysis of AI systems and for the advancement of the technical and philosophical understanding of the notions of Bias, Risk and Opacity in AI, with the ultimate objective of generally contributing to the development of trustworthy AI.[1]

The aim of the collaboration between BRIO and Alkemy is to produce software applications for the analysis of bias, risk and opacity with regard to AI technologies which often rely on non-deterministic computations and are often opaque in nature. One of the most challenging aspects of modern AI systems, indeed, is that they do not guarantee specification correctness, and they are not transparent, in the sense that a general formal description of their behaviour might not be available.

We present a first tool developed within the BRIOxAlkemy collaboration for the detection and analysis of biased behaviours in AI systems. The tool is aimed at developers and data scientists that wish to test their algorithms relying on probabilistic and learning mechanisms in order to detect misbehaviours related to biases and collect data about them. The ultimate goal is to provide them with useful insights and data for the improvement of the AI systems with respect to bias.

BEWARE23: Workshop on Bias, Risk, Explainability and the role of Logic and Logic Programming, 6–9 November 2023, Rome, Italy

*Work funded by the PRIN project n.2020SSKZ7R BRIO (Bias, Risk and Opacity in AI).

*Corresponding author.

greta.coraglia@unimi.it (G. Coraglia); fabio.dasaro@univr.it (F. A. D'Asaro); francesco.genco@unimi.it (F. A. Genco); davide.posillipo@alkemy.com (D. Posillipo); giuseppe.primiero@unimi.it (G. Primiero); xxx@xxx (C. Quaggio)

CEUR Workshop Proceedings (CEUR-WS.org)

[1]sites.unimi.it/brio

The system presented is based on the theoretical, foundational works presented in [1], [2], and [3], and in this paper we discuss the formal ideas behind its development (Section 2), the technical details of its implementation (Section 3), the data obtained from a case study that validates the usefulness of the software (Section 4), and the projected further developments aimed at integrating the presented software in a more general tool for the analysis of AI systems (Section 5).

2. Theory

Our tool takes as input an AI model's output and a set of user's selected parameter settings: the former is encoded as a set of datapoints with their associated features, the latter include the designation of a sensitive feature. The output is an evaluation of the possibility that the AI model under consideration is biased with respect to the features designated as sensitive by the user. The system closely guides the user in the process of setting the parameters and provides it with detailed explanations, remaining customisable with respect to the mathematical details of the analysis. The choices left to the user are those that actually make a conceptual difference in the outcome of the analysis and this difference is explained to the user during the setting of these parameters.

The analyses conducted by our system are of two kids:

1. comparison between the behaviour of the AI system and a target desirable behaviour, and
2. comparison between the behaviour of the AI system with respect to a sensitive class $c_1 \in F$ and the behaviour of the AI system with respect to another sensitive class $c_2 \in F$ related to the same sensitive feature F.

The second comparison includes a subsequent check, in case we find a biased behaviour, on some (or all) of the subclasses of the considered sensitive classes. This second check is meant to tell us if the bias encountered at the level of the classes can be explained away by other features of the individuals in the sensitive classes considered that are not related to the sensitive feature at hand.

Before we delve into the technical details, let us give a little example: given a database of individuals, with their age, gender, and level of education detailed, consider an algorithm which tries to predict whether they are likely to default on credit. We wish to check if age is a sensitive factor in such prediction. We feed the tool our dataset, the output of the run of the predictive algorithm, and mark the feature of age as sensitive. The program allows us to compare either how the behaviour of the algorithm with respect to age differs from an "optimal" behaviour (in this case, we might consider optimal the case where each age group equally succeeds), or how different age groups perform with respect to one another.

2.1. Divergences

In order to compute the difference between the behaviour of the AI system under investigation – which we identify with the probability distribution Q – and an optimal behaviour P, in case some data about such an optimal behaviour is available, we might employ different divergences.

A first possibility is a divergence borrowed from information theory: The Kullback–Leibler divergence D_{KL} [4]. This measure is mathematically expressed by the following equation

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in X} P(x) \cdot \log\left(\frac{P(x)}{Q(x)}\right)$$

and it intuitively indicates the difference, in probabilistic terms, between the input-output behaviour of the AI system at hand and a reference probability distribution. Before adding up all the the differences computed for each possible output of the AI system, these differences are weighted by the actual probability of correctly obtaining that output.

Since the divergence D_{KL} is not symmetric, when we have to compare the behaviour of the system on a sensitive class with the behaviour of the system on another sensitive class, we must adopt another measure. We could either simply adopt a symmetric version $D_{\mathrm{KL}}^{\mathrm{Sym}}$ of D_{KL},

$$D_{\mathrm{KL}}^{\mathrm{Sym}}(P \parallel Q) = \frac{D_{\mathrm{KL}}(P \parallel Q) + D_{\mathrm{KL}}(Q \parallel P)}{2},$$

or go for the very simple but intuitive D_{TV}

$$D_{\mathrm{TV}}(P \parallel Q) = \sup_{x \in X} |P(x) - Q(x)|,$$

which computes the greatest difference between the probability of obtaining an output o when we run our AI system on the individuals of the sensitive class $c_1 \in F$ and the probability of obtaining the same output o when we run our AI system on the individuals of the sensitive class $c_2 \in F$.

Finally, in order to normalise the values of the different divergences that we use in the system and be able to select common, reasonable thresholds for all of them, we project the values of D_{KL} and $D_{\mathrm{KL}}^{\mathrm{Sym}}$ – which are not between 0 and 1 by default – onto the interval $[0, 1]$ as follows: instead of $D_{\mathrm{KL}}(P \parallel Q)$ and $D_{\mathrm{KL}}^{\mathrm{Sym}}(P \parallel Q)$, respectively, we use

$$1 - \frac{1}{D_{\mathrm{KL}}(P \parallel Q)} \quad \text{and} \quad 1 - \frac{1}{D_{\mathrm{KL}}^{\mathrm{Sym}}(P \parallel Q)}\,.$$

2.2. How to handle sensitive features corresponding to more than two classes

Since the divergences the system uses are binary functions, it is not obvious how to handle the case in which the sensitive feature we are considering partitions the domain in three or more classes. It is, indeed, easy to compute the divergence for pairs of classes, but we also need, in this case, a sensible way of aggregating the obtained results in order to obtain one value describing well how the AI model behaves with respect to the sensitive feature under consideration. It turns out that there are several ways of doing this and each of them tells us something of value if we wish to reach a meaningful conclusion about the AI model.

Suppose that we are studying the behaviour of the model with respect to the feature $F = \{c_1, \ldots, c_n\}$ which induces a partition of our domain of individuals into the classes $c_1, \ldots, c_n$. The first step, required in any case, is the pairwise calculation of the divergences with respect

to the different classes induced by F. Hence, for each pair $(c_i \parallel c_j)$ such that $c_i, c_j \in F$, we compute $D(c_i \parallel c_j)$ where D is the preselected divergence and consider the set $\{D(c_i \parallel c_j) : c_i, c_j \in F \& i \neq j\}$. For instance, if we are considering age as our feature F and we partition our domain into three age groups, we might have

$$F = \{over\_50\_yo, between\_40\_and\_50\_yo, below\_40\_yo\}$$

We can then use the following ways of aggregating the obtained values.

The maximal divergence $\max(\{D(c_i \parallel c_j) : c_i, c_j \in C \& i \neq j\})$

The maximal divergence corresponds to the worst case scenario with respect to the bias of our AI system. This measure indicates how much the AI system favours the most favoured class with respect to the least favoured class related to our sensitive feature.

The minimal divergence $\min(\{D(c_i \parallel c_j) : c_i, c_j \in C \& i \neq j\})$

The minimal divergence, on the other hand, corresponds to the best case scenario. This measure indicates the minimal bias expressed by our AI system with respect to the sensitive feature. If this measure is 0, we do not know much, but if it is much above our threshold, then we know that the AI system under analyis expresses strong biases with respect to all classes related to the sensitive feature.

The average of the divergences $\sum_{c_i, c_j \in C} D(c_i \parallel c_j) / \binom{|C|}{2}$

That is, the sum of the divergences between the two elements of each pair $c_i, c_j \in C$ divided by the total number $\binom{|C|}{2}$ of these pairs. This measure indicates the average bias expressed by the AI system. Unlike the previous measures, this one meaningfully combines information about the behaviour of the system with respect to all classes related to the sensitive feature. What this measure still does not tell us, though, is the variability of the behaviour of the system in terms of bias. The same average, indeed, could be due to a few very biased behaviours or to many mildly biased behaviours.

The standard deviation of the divergences

$$\sqrt{\left(\sum_{c_i, c_j \in C} (D(c_i \parallel c_j) - \mu)^2 / \binom{|C|}{2} \right)}$$

where $\mu = \sum_{c_i, c_j \in C} D(c_i \parallel c_j) / \binom{|C|}{2}$. That is, the square root of the average of the squares of the divergences between each value $D(c_i \parallel c_j)$ and the average value of $D(c_i \parallel c_j)$ for each pair c_i, c_j. In other terms, we calculate the average of the divergences, then the difference of each divergence with respect to their average, then we square this differences and calculate their average. Finally, we compute the square root of this average. This measure indicates the variability of the bias expressed by the AI system we are inspecting. That is, how much difference there is between the cases in which the system is behaving in a very biased way and the cases in which the system is behaving in a mildly biased way. This information complements the information we can gather by computing the previous measure.

2.3. Parametrisation of the threshold for the divergence

Most of the times the system computes a divergence between the results of the model and a reference distribution or the results of the model for one class and another class, a threshold is employed to check whether the divergence is significant—and should be treated as a possible violation of our fairness criteria—or negligible—and thus should be simply ignored. Obviously, fixing this threshold once and for all would give a rather inflexible system.

For instance, setting the threshold to $1/100$ might be reasonable if we are considering the results of the model on a set of 100 individuals, but it is clearly too strict if our domain only contains 40 individuals. Indeed, in the latter case, even a difference only concerning one individual would constitute a mistake much greater than the one admitted by the threshold.

This is why we parametrise the threshold on the basis of several factors. First, the user has to decide how much rigour is required when analysing the behaviour of the model with respect to the feature indicated as sensitive for the present analysis. Two settings are possible: `high` and `low`. The setting `high` implies that the considered feature is very sensitive and that the system should be extra attentive about the behaviour of the model in relation to it, while `low` implies that differences in the behaviour of the model with respect to the considered feature are significant only if they are particularly strong or extreme. This setting distinguishes between a thorough and rigorous investigation and a simple routine check, one might say.

The second factor considered in computing the threshold is the number of classes related to the sensitive feature under consideration. We could call this the *granularity* of the predicates related to the sensitive feature, and it indicates how specific the predicates determining the studied classes are. Specific predicates describe in more detail the objects of our domain and determine more specific classes. When the classes that we are considering are very specific, the divergence generated by the bias of the model can be spread over several different classes. Hence, we need to be more attentive about the small divergences that appear locally—that is, when we check pairs of these classes. The threshold should thus be stricter.

Finally, each time we compute a divergence relative to two classes, we scale the threshold with respect to the cardinality of the two classes. Large classes mean a stricter threshold. This is a rather obvious choice related to the fact that statistical data related to a large number of individuals tend to be more precise and fine grained. We already gave an example of this few lines ago.

Technically, the threshold is always between 0.05 and 0.005, the setting `high` restricts this range to the interval $[0.005, 0.0275]$ and the setting `low` to the interval $[0.0275, 0.05]$.

The number n_C of classes related to the sensitive feature and the number n_D of individuals in our local domain (that is, the cardinality of the union of the two classes with respect to which we are computing the divergence) are used to decrease the threshold (in other words, to make it stricter) proportionally with respect to the interval selected. The greater the number n_C, the smaller the threshold, the greater the number n_D, again, the smaller the threshold. The threshold is then computed as

$$\epsilon = (n_C \cdot n_D) \cdot m + (1 - (n_C \cdot n_D)) \cdot M$$

where m is the lower limit of our interval and M is its upper limit.

3. Implementation

@Alkemy

4. Validation

@Alkemy

5. Conclusions and Further Work

@UNIMI

The presented tool for the detection of biases in AI systems is meant to be a module integrated into a more complex software application consisting of three, partly inter-dependent modules. The complete system will present two autonomous modules for the analysis of bias and opacity of AI systems—the first of which we just presented—and a module depending on these two concerning risk evaluation.

Acknowledgments

This research has been partially funded by the Project PRIN2020 "BRIO - Bias, Risk and Opacity in AI" (2020SSKZ7R) and by by the Department of Philosophy “Piero Martinetti” of the University of Milan under the Project “Departments of Excellence 2023-2027”, both awarded by the Ministry of University and Research (MUR).

References

[1] F. D’Asaro, G. Primiero, Probabilistic typed natural deduction for trustworthy computations, in: Proceedings of the 22nd International Workshop on Trust in Agent Societies (TRUST2021@ AAMAS), 2021.

[2] F. D’Asaro, F. Genco, G. Primiero, Checking trustworthiness of probabilistic computations in a typed natural deduction system, ArXiv e-prints (2022).

[3] F. Genco, G. Primiero, A typed lambda-calculus for establishing trust in probabilistic programs, ArXiv e-prints (2023).

[4] S. Kullback, R. A. Leibler, On Information and Sufficiency, The Annals of Mathematical Statistics 22 (1951) 79 – 86. URL: https://doi.org/10.1214/aoms/1177729694. doi:10.1214/aoms/1177729694.

A. Online Resources

The tool.