Received April 1, 2020, accepted May 17, 2020, date of publication May 21, 2020, date of current version June 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2996495

Uncertainty-Based Rejection Wrappers for Black-Box Classifiers

JOSÉ MENA[1,2], ORIOL PUJOL[2], AND JORDI VITRIÀ[2]
[1]Eurecat, Centre Tecnològic de Catalunya, 08290 Barcelona, Spain
[2]Departament de Matemàtiques i Informàtica, Universitat de Barcelona, 08007 Barcelona, Spain

Corresponding author: José Mena (jose.mena@eurecat.org)

This work was supported in part by the Spanish Project under Grant TIN2016-74946-P, Grant RTI2018-095232-B-C21 (MINECO/FEDER, UE), and Grant 2017SGR1742, and in part by the AGAUR of the Generalitat de Catalunya through the Industrial Ph.D. Grant Program.

ABSTRACT Machine Learning as a Service platform is a very sensible choice for practitioners that want to incorporate machine learning to their products while reducing times and costs. However, to benefit their advantages, a method for assessing their performance when applied to a target application is needed. In this work, we present a robust uncertainty-based method for evaluating the performance of both probabilistic and categorical classification black-box models, in particular APIs, that enriches the predictions obtained with an uncertainty score. This uncertainty score enables the detection of inputs with very confident but erroneous predictions while protecting against out of distribution data points when deploying the model in a productive setting. We validate the proposal in different natural language processing and computer vision scenarios. Moreover, taking advantage of the computed uncertainty score, we show that one can significantly increase the robustness and performance of the resulting classification system by rejecting uncertain predictions.

INDEX TERMS Classification, machine learning, rejection systems, uncertainty.

I. INTRODUCTION

Recently, the progress on Artificial Intelligence, fostered by the advent of technologies like Deep Learning (DL) and Big Data technologies, is leading to extraordinary results that even outperform humans in some tasks. Thanks to the availability of humongous volumes of data and sophisticated software architectures, companies are being able to release high-performance prediction models by leveraging advanced technologies such as computer vision or natural language processing.

Sometimes, these pre-trained models are made publicly available in a form that allows their ready application to a target domain after some fine-tuning. However, many times, companies develop these models as part of their machine learning-as-a-service (MLaaS) strategy, including them in their portfolio of APIs. In this case, customers are only able to query these APIs for their particular use case and data. The resulting predictions can then be integrated into their final products as part of a more complex solution.

Because MLaaS is becoming a sensible option for non-technological companies, some points must be considered before integrating APIs or third-party libraries into their solution. First, as customers, in most of the cases, they do not have either access to the internals of those models nor to the data used to train them. It is conceivable that the data used for training significantly differ from the target domain and, despite the generalization power of DL models, this domain shift may have a significant impact on the ultimate predictive power. For example, it might happen that the API does not fit exactly to the target problem. Maybe the NLP model was trained for a different variation of language, or maybe the output categories of a visual classification model do not precisely match the type of images present in the target domain.

There are many scenarios where the potential risks of applying a pre-trained model to a different domain might lead to unpredictable or harmful consequences. We can consider, for example, the cases of medical diagnosis [1] or self-driving cars [2]. This paper describes a method to account for the accuracy risk of existing classification systems, identifying the quality and reliability of the predictions in the target domain. In particular, the present work focuses on the problem of using black-box models, such as pre-trained language embedding models, third-party visual recognition

The associate editor coordinating the review of this manuscript and approving it for publication was Paolo Remagnino.

models, or in general, trained models that are served as APIs. We suppose that the internals of the model are not disclosed or are obfuscated to the practitioner.

In particular, when in front of high-stakes decisions, it is usually better to weight the decision according to the certainty and confidence than to assume hard predictions for all cases. The goal of the present paper is to present a method for estimating the uncertainty of a black-box model. The presented method allows us to assess what the model does not know when applying the black-box system to a target application and to propose a rejection mechanism based on the uncertainty associated with each prediction.

More specifically, we introduce a deep learning wrapper, based on the Dirichlet distribution, that given any black-box classification system allows the measurement of three different uncertainty sources, namely:

- uncertainty associated with cases that are close to the decision boundary,
- uncertainty associated with cases that, while being far from the decision boundary, get overconfident but erroneous predictions,
- uncertainty associated with out of source distribution cases.

These measures may serve as a prophylactic measure against the possible changing environment or distribution shifts in the data when the system is deployed. Moreover, we prove that the proposed method is valid for multinomial output category distributions as well as for hard categorical classification systems, where the output is a single category value.

We additionally show that the uncertainty score obtained employing the proposed wrapper is a suitable metric for rejection techniques. The wrapper improves the state-of-the-art in the cases of overconfident wrong predictions and out-of-distribution samples. It provides competitive results in the case of samples that are close to the decision boundary.

The wrapper is validated in a large variety of problems, including natural language processing and computer vision problems. Besides showing ablation studies on the parameters of the wrapper, we discuss the practical use of the wrapper when the black-box and the target domain have a different number of classes, or even new classes, and compare it with calibrated methods.

The rest of this paper is structured as follows. In Section 2, we review previous works related to the estimation of uncertainty in DL and its usage in rejection systems. Section 3 introduces the wrapper proposed for modelling the uncertainty through a Dirichlet distribution and in section 4, we describe how to use this metric for rejecting uncertain prediction, thus increasing the performance of the black-box classifiers as applied to the target domain. Section 5 motivates the method proposed with a toy scenario. In section 6, we describe the two scenarios employed for illustrating the proposed method and validate its effectiveness and section 7 holds the results obtained, including an ablation study of different alternatives in the design of the wrapper. Finally, section 8 concludes the article.

II. RELATED WORK

This section includes an overview of previous works that have explored the role of uncertainty in deep learning models and different strategies for estimating it. Next, we survey the literature about classification with rejection methods and discuss how to measure their performance.

A. UNCERTAINTY MODELLING METHODS

In machine learning in general and deep learning classifiers in particular, two main types of uncertainty have been considered: epistemic and aleatoric. **Epistemic uncertainty** corresponds to the uncertainty associated with the model. This is to which extent the training data determine our model. In principle, we could reduce this type of uncertainty by training the model with more data. In the black-box setting studied in this paper, this option is not possible, as we do not have access to the model internals. The other type of uncertainty, **aleatoric uncertainty**, is related to the data measurement process and can not be reduced by adding more samples to the training process. When referring to this last type of uncertainty, we can consider two different scenarios:

- Homoscedastic uncertainty, when the level of noise derived from the measurement process remains constant for all the data samples.
- Heteroscedastic uncertainty, when the level of noise derived from the measurement process depends on data samples.

Different types of uncertainty must be measured differently. Given a training data set, D, composed by pairs of data points and their corresponding labels, $D = \{(x_i, y_i)\}, \quad i = 1 \ldots N$, and given a new sample x^*, our task is to predict its corresponding label y^*, assuming a parametric model with parameters W (corresponding to the black-box). Our goal can be understood as the estimation of the conditional distribution of the outputs:

$$p(y^*|x^*, D) = \int_W p(y^*|x^*, W)p(W|D)dW \tag{1}$$

Eq.1 shows the origin of both uncertainty terms. $p(y^*|x^*, W)$ depends on the application of the model to the input data, while $p(W|D)$ is measuring how the model parameters depend on the training data. Thus, the first term is modelling the aleatoric uncertainty, as it measures how the output is affected by the input data given a model. Furthermore, the second term is modelling the epistemic uncertainty as it measures the uncertainty induced by the parameters of the model.

In the literature of deep learning, we can find different approaches for modelling the uncertainty in classification systems. Most of them are focused on epistemic uncertainty (they focus on the term $p(W|D)$ from Eq.1), and they try to estimate the probability distributions of the parameters of the model. In [3], authors present a probabilistic backpropagation approach for training Bayesian neural networks. In those networks, model parameters are modeled as probability distributions, making them account for uncertainty in parameters and predictions naturally. In [4], authors propose a more straightforward method for inducing variability in the parameters by introducing a Monte Carlo Dropout as a proxy for the probabilities of the weights. On prediction time, they use the variance of the predictions to obtain a measure of uncertainty. In [5], authors apply Monte Carlo Dropout for computing the uncertainty in classification problems on computer vision. All these works propose different ways for accounting for epistemic uncertainty by introducing elements or changing the model internals. However, in the present work, we are dealing with black-box models. As such, we can neither alter the definition of the model nor access to the internals. The impossibility of altering the parameters of the black-box model prevents the estimation of epistemic uncertainty.

Thus, focusing on the heteroscedastic aleatoric uncertainty, we assume a given model with parameters W, and we try to model the probability distribution on the output $p(y^*|W, x^*)$. Kendall at al. [5] proposed a way to compute aleatoric uncertainty by modelling the logits of the output as independent Gaussian random variables. They also proposed the use of different metrics to measure uncertainty [6]. Alternatively, and based on the seminal work of Fernandes and Oliveira [7], some recent papers [8], [9] proposed the use of the Dirichlet distribution for modelling the output of the network. Additionally, some authors have proposed the use of Dirichlet output distributions for detecting out-of-sample data points [10], [11]. However, these works present two main issues that prevent them from being applied in the black-box setting. First, they need to have access to the internals of the model in order to inspect its weights or to modify the model architecture. Second, the uncertainty model is learned together with the classifier. These characteristics prevent the use of these approaches in black-box models.

Alternatively to the commented approaches where uncertainty is measured by accessing the original model, we can also consider uncertainty metrics that can be directly derived from the discrete probability distribution of the output of the classifier. Different metrics can be inferred from this distribution, such as v. It is worth noting that these methods can only capture one kind of uncertainty: that related to the samples that are close to the boundary. As we show in the experimental section, our method can capture a richer set of uncertainty types and also allows for computing uncertainty in hard decision scenarios.

B. CLASSIFICATION WITH REJECTION TECHNIQUES

The process of abstaining on producing an answer or discarding a prediction when the system is not confident enough can be found under different names in the machine learning literature: Rejection Methods, Selective Classification, Abstention, or Three-Way Classification. The most common approach sets a threshold based on a given metric and discards predictions accordingly. Following the ideas proposed in the seminal work of [12], where the authors set a threshold for rejection with which to minimise the classification risk, other works like [13]–[16] consider different cost-based rejection models based on the output responses of deep learning models.

Alternatively, other works [14], [17], [18] include a term for the rejector in the loss that is learned together with the classifier. Following a similar approach, in [19], the authors propose to omit noisy labels in order to improve the training process. These works have in common the approach of considering a fixed-cost for the abstention, in which the classifier incurs a fixed cost every time the abstain option is invoked. On the contrary, in [20], authors set a fixed fraction of input samples, δ, in which the learner is allowed to abstain without incurring any costs. They propose a plug-in classifier that employs unlabelled samples to decide the region of abstention and derive an upper-bound on the excess risk. Another strategy is followed by authors of [21], where they propose to train the model together with a particular loss function for rejection. In this case, though, they fit a model specifically for each degree of coverture chosen a priori.

The main limitation of the methods described so far is the fact that they jointly learn the classification model together with the rejection function. Unfortunately, in a black-box scenario, re-training the model is not an option.

Finally, the evaluation of the performance of classification with rejection models usually use standard metrics such as accuracy or F1-score for obtaining accuracy-rejection curves (ARC) [22] or 3D receiver operating characteristic [23]. Other works focus on the rejection of multi-label classification systems [24]. These approaches present limitations as they are not able to determine the optimal rejection rate by comparing the performance of the classifiers. Beyond the ROC space, some authors [25] have analysed different representation spaces to build a rejection system, mainly the cost-reject (CR) and the error-reject (ER) space. The first one adapts better to the case we are considering in this work, where we do not have fixed cost criteria. Nonetheless, in this article, we follow the approach introduced in [26], where authors propose three different performance measures for evaluating the best rejection point that overcomes the previous restrictions, namely non-rejected accuracy, and classification and rejection quality.

III. MEASURING UNCERTAINTY IN BLACK-BOX SYSTEMS

Our goal is to estimate the uncertainty in the predictions of a black-box model. Thus, we propose a wrapper algorithm that takes a black-box model and operates on top of it. As such, there are several constraints to observe. First, we need to exclusively operate on the inputs and outputs of the black-box classifier. We are not allowed to use any intermediate or internal value of the black-box model. Second, the input of the wrapper has to be compatible with the original distribution over the output classes. Finally, the wrapper must be able to operate on both probabilistic and categorical classification systems.

As we have shown in the literature review, there are different ways of modelling the uncertainty in deep learning models. In [5], [27], authors use independent Gaussian random variables to model the pre-activation value of the logits and Monte Carlo sampling to obtain the variance of the final output. In our opinion, independent Gaussian distributions impose unnecessary assumptions and need for additional normalization steps. Here we consider a more natural approach for the output distribution and suppose it can be modelled by a Dirichlet probability distribution, following [8], [10], [11]. Again, the problem with these approaches is that they do not conform to the black-box constraints here imposed, as they need to train the model and to have access to internal parameters. We propose a new method that combines a Dirichlet output distribution with a Monte Carlo sampling method that allows for modelling the uncertainty while observing the black-box restrictions. This approach results in a model that is compatible with both hard and soft predictions.

A. DIRICHLET CONCENTRATION REPARAMETERIZATION

We denote with $\mathcal{D}_{target}$ the data set of our target application. This is the data corresponding to the domain we want to apply the black-box classifier, f^{bb}. $\mathcal{D}_{target}$ is composed of pairs $\{(x_i, y_i)\}_{i=1\ldots N}$, where $y_i \in \mathbb{R}^{C_{target}}$ and C_{target} is the number of different classes. This data set is usually different from that used to train the black-box model. When applying the black-box model to our data, we obtain a new prediction $\hat{y}_{target} = f^{bb}(x_{target})$.

Sometimes, the original black-box model may have been trained with a slightly different set of target labels, C_{bb}, than our domain. Thus we also consider a mapping function m that maps the classes from one set to the other, $m(y) : C_{bb} \rightarrow C_{target}$.

A wrapper is defined as a model, f^w, that considers a black-box and operates on its inputs and outputs while potentially adding new features:

$$y^w = f^w(f^{bb}(x), x). \tag{2}$$

Initially, we will consider the output of the black-box classifier, $f^{bb}(x)$, to be a vector representing the multinomial distribution associated to the probabilities of belonging to each of the output classes. Later we will relax this assumption and consider pure hard categorical outputs. The proposed wrapper changes the black-box output vector into a random variable that follows a Dirichlet distribution, modelling, therefore, the probability of the multinomial black-box output where w are the parameters of the wrapper:

$$p(y^w|x, m(y^{bb}), w) \sim Dir(\boldsymbol{\alpha}), \tag{3}$$

We will also impose the wrapper to preserve the translated multinomial output $m(y^{bb})$ of the black-box in expectation:

$$\mathbb{E}(y^w) = m(y^{bb}). \tag{4}$$

We propose to use a decomposition of the concentration parameter in two terms to relate the output of the black-box classifier,[1] y^{bb}, with the concentration parameter, $\boldsymbol{\alpha}$, in the Dirichlet distribution of the wrapper. To that effect, we recall some basic statistics of the Dirichlet distribution.

Given a Dirichlet random variable $x \in \mathbb{R}^C$ with concentration parameter $\boldsymbol{\alpha} \in \mathbb{R}^C$, the expected value of the distribution is defined as $\mathbb{E}(\boldsymbol{x_i}) = \boldsymbol{\alpha_i}/\sum_{k=1}^{C} \alpha_k$.

Observe that the expected value has the same properties as a probability distribution and that the output of the black-box $y^{bb} \in \mathbb{R}^C$ can also be regarded as a probability distribution. In this sense, we could directly use the of the black box as the concentration parameter. However, by taking advantage from the fact that each term of the concentration parameter is not necessarily constrained to the interval [0, 1], we introduce a new scalar parameter, $\beta \in \mathbb{R}$ that allows the wrapper to adapt the distribution properties to the input data:

$$\boldsymbol{\alpha} = \beta y^{bb}. \tag{5}$$

Introducing this parameter fulfills the constraint in Eq.4, i.e. $\mathbb{E}(y^w) = y^{bb}$.

By means of this decomposition, while the output of the black-box classifier stands for the mean, parameter β defines the shape of the distribution. This approach share similarities to the decompositions present in other works in a different context[2] [8], [10], [11].

This decoupling allows to effectively isolate the contribution of the black-box and the value of parameter β. Figure 1 shows the integration of the wrapper (in light green colour) with the black-box classifier (in grey). Observe that the wrapper consists of two blocks. First, the Dirichlet reparameterization layer of the wrapper that decouples the influence of the black-box model from the rest (see the dashed line). Then, a deep learning architecture[3] which aims to compute the scalar value of β.

Moreover, the figure also shows a possible decomposition of the most usual building blocks:

- A latent representation block that adapts the to the nature of the input data (e.g. recurrent models or embeddings for natural language processing, or convolutional blocks for image data).
- A standard feedforward/convolutional block that squeezes the information into the single output parameter β.

B. INFERENCE IN THE DIRICHLET SETTING

Similarly to [5], we approximate the expected value of the classification probabilities using Monte Carlo sampling. The main difference here is the fact that we sample the learned Dirichlet distribution to directly obtain the output instead of

[1]For the sake of simplicity. Without loss of generalization, in this section, we assume an identity m function so that $m(y^{bb}) = y^{bb}$

[2]It is worth noting that in the context of those works, there is a degradation in performance when using Dirichlet. This degradation does not happen in our case since the black-box model is non-mutable.

[3]The architecture used in this figure corresponds to the one used in the experimental section.

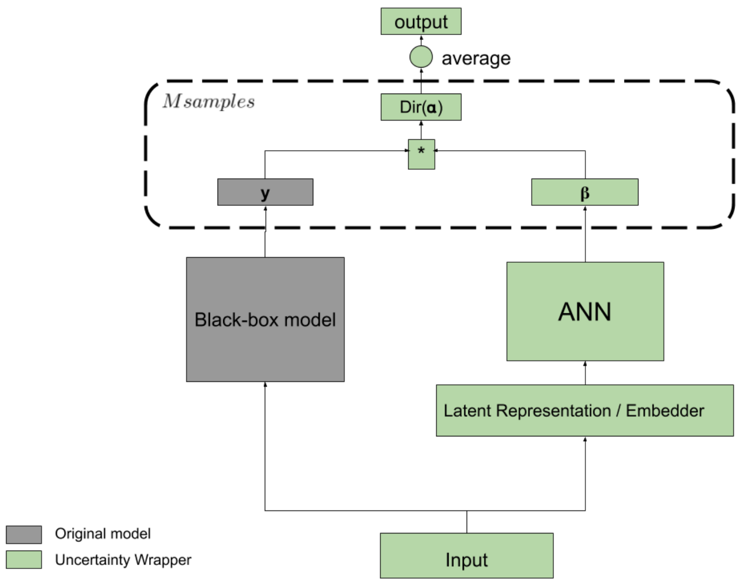


FIGURE 1. **Model used to estimate the aleatoric uncertainty from the original black-box model.**

each of the logits individually for each sample, as follows, where M is the number of times we sample from the Dirichlet distribution:

$$y^w \sim Dir(\alpha), \quad \mathbb{E}[y^w] = \frac{1}{M}\sum_{m=1}^{M} y_m^w, \tag{6}$$

This expectation, obtained from the samples drawn from the resulting Dirichlet distribution, defines the outcome of the model. During the learning phase, we use the minimization of a loss function, that in the present work is split into two terms. The goal of this separation is to use an alternating learning schema that will allow to train the estimation of the uncertainty for the target training set and also model out-of-distribution points. For the former, we use a regularized version of the cross-entropy loss function [28]:

$$\begin{aligned}\mathcal{L}_{ale}(W) &= -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{C}\sum_{c=1}^{C} y_{i,c}\log \mathbb{E}[y_i^w]_c + \lambda\|\beta\|_2 \\ &= -\frac{1}{N}\frac{1}{C}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log\Big(\frac{1}{M}\sum_{m=1}^{M} y_{m,i,c}^w\Big) + \lambda\|\beta\|_2\end{aligned} \tag{7}$$

Observe that we introduce the norm of the β value in the minimization function. This term is required since the unregularized cross-entropy forces the value of β to grow unbounded. By adding this term, we control its growth and govern the trade-off with a scalarization parameter λ.

The second term of the loss minimizes the Kullback-Leibler divergence between the expected output and a Uniform distribution over the labels [29]:

$$\mathcal{L}_{ood}(W) = \mathcal{KL}(\mathcal{U}(y)\|\mathbb{E}[y^w]) = \mathcal{KL}(\mathcal{U}(y)\|\Big(\frac{1}{M}\sum_{m=1}^{M} y_m^w\Big) \tag{8}$$

Both losses are combined into the final training loss as follows,

$$\mathcal{L}(W) = (y_o)L_{ale}(W) + (1-y_o)L_{ood}(W) \tag{9}$$

where, in order to enable the alternating learning scheme when modelling the out-of-distribution, we include an additional label y_o that is used to change among both losses and identifies whether the sample corresponds to the target data set or to an additional set that is modelling out-of-distribution. To this effect, the original training set gets augmented with a set of inputs that are intentionally out of the original distribution. For example, in natural language processing, the added elements might belong to a domain different from the target problem. In computer vision, they might be images from categories not present in the target application. The label, y_o, will tell whether the input belongs or not to the distribution:

C. OBTAINING AN UNCERTAINTY SCORE FROM THE WRAPPER

Modelling the output of the black-box model with the described Dirichlet layer allows studying the variability of the parameters of the black-box output distribution. Using Monte Carlo simulation, we can characterize the heteroscedastic aleatoric uncertainty. Similarly to what Gal *et al.* did in [6] with the Gaussian variables associated with the logits, in this case, we can directly sample from the Dirichlet distribution.

Then, we can use standard techniques for measuring uncertainty like *variation ratios* or *predictive entropy*.

Variation ratios measures the variability of the predictions obtained from sampling [30] by computing the fraction of samples with the correct output. This heuristic is a measure of the dispersion of the predictions around its mode. To that respect, for a given data point x we can sample M output vectors from the Dirichlet wrapper $y_x^t, t = 1 \ldots M$, and compute the variation ratios as follows,

$$VR = 1 - \frac{f_{c^*}}{M}$$

where $f_{c^*} = \sum_t 1[y_x^t = c^*]$, and c^* corresponds to the sampled majority class,

$$c^* = \arg \max_{c=1,\ldots,C} \sum_{t=1}^{M} 1[y_x^t = c]. \tag{10}$$

Alternatively, *predictive entropy* considers the average amount of information contained in the predictive distribution. In our case, this corresponds to the vectorial output of the black-box. Results with low entropy values correspond to confident predictions, whereas high entropy leads to large uncertainty. Since the output of the black-box model y^m already describes a probability distribution, one could compute its predictive entropy and obtain a measure of its uncertainty with

$$\mathbb{H} = -\sum_c y_c^m \log y_c^m$$

However, as the wrapper allows us to model the variability of the parameters of the black-box output distribution, we can compute a different score based on the predictive entropy. This score takes into account the variability of the predicted value by sampling from the wrapper inferred distribution, the sampled predictive entropy, defined as

$$\mathbb{H} = -\sum_c \mathbb{E}[\hat{y}]_c \log \mathbb{E}[\hat{y}]_c. \tag{11}$$

As we show in the experimental section, this latter approach better captures the uncertainty compared to the rest of the methods.

IV. CLASSIFICATION WITH REJECTION SYSTEM

So far, in this article, we have presented a way to model uncertainty in single-label classification problems. In this section, we propose a rejection system for this type of classification problems that takes advantage of the measured uncertainty. In a rejection setting, the model is enhanced with a new class, the *rejection* class. Thus, the final prediction becomes

$$\hat{y} = \begin{cases} f(x), & \text{if } \phi(x) < \tau \\ reject, & \text{if } \phi(x) \geq \tau \end{cases}$$

Here, $f(x)$ is the classifier without rejection, the function $\phi(x)$ is a function on the input that evaluates the confidence of the prediction model, and τ is a threshold value that indicates the rejection point. Those predictions with lower confidence than the given threshold are rejected, whereas those more confident are predicted using the original classifier.

As mentioned in the review of related work, in the context of the present work, the $\phi(x)$ function can not be learned together with the classifier when it comes to pre-trained black-boxes. In this case, we must obtain the confidence score from the input and the corresponding prediction. In probabilistic classifiers, risk can derive from the observation of the output probabilities employing different metrics, like Least Confidence [31], Margin of Confidence [32] and Variation Ratios and Predictive Entropy, as proposed in [6].

In the present work, we propose to use the entropy obtained from the wrapper defined in Eq.11 as the $\phi(x)$ function to model the uncertainty of black-box classification systems. To validate that the proposed method better captures the uncertainty in such models, we apply the performance measures proposed in [26]. In order to evaluate the rejection metric, we split the dataset using two criteria: whether the method **R**ejects the data point or **N**ot; and whether the point is **A**ccurately classified,[4] or **M**issclassified named as R, N, A or M respectively. Using this terminology, we follow the guidelines in [26] for rejection quality metrics. Using these values, three quality metrics can be derived (and illustrated in Fig.2):

Non-rejected

- **Accuracy** measures the ability of the classifier to classify non-rejected samples accurately. It is computed as follows,

$$NRA = \frac{|A \bigcap N|}{|N|}$$

- **Classification Quality** measures the ability of the classifier with rejection to classify non-rejected samples accurately and to reject misclassified samples. It is computed as follows,

$$CQ = \frac{|A \bigcap N| + |M \bigcap R|}{|N| + |R|}$$

- **Rejection Quality** measures the ability to concentrate all misclassified samples onto the set of rejected samples. It is computed as follows,

$$RQ = \frac{|M \bigcap R|\, |A|}{|A \bigcap R|\, |M|}$$

An optimal rejection point will show a trade-off between the three metrics, being able to divide misclassified predictions from the right ones and preserve only those points that provide useful information. The higher the value displayed, the better that metric performs for rejection. An option to automatize the selection of the rejection point could be to use a method like [16], where they use a binary search to optimize it.

V. ILLUSTRATION OF THE DIRICHLET WRAPPER AND UNCERTAINTY MEASUREMENT

In this section, we introduce a toy scenario to illustrate the intuition of the proposed method. Recall that we are dealing with a black-box, single-label probabilistic classifier. The outcome of this system consists of a probability distribution of C different possible categories. The original black-box was trained to match the categorical distribution of the labels in an unknown source domain.

[4]We measure accuracy by comparing the test label with the category that holds a higher probability

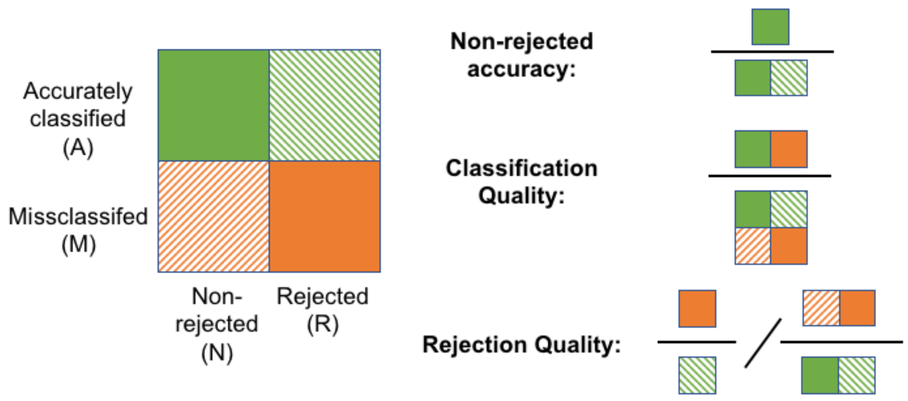


FIGURE 2. Rejection performance metrics as proposed in [26].

Here we propose to model the output of the black-box as a Multinomial distribution. In this section, we show how, by considering the output probabilities as generated by the Dirichlet probability distribution and not as single-point estimates, we can model more precisely different sources of uncertainty.

We generate a total of 20000 two dimensional points corresponding to three different classes, each of them modelled as a Gaussian distribution (Figure 3(a)). These points are split into train, validation and test sets. We randomly select 9600 points for training a classification model for the three classes. The accuracy of the resulting classifier reaches 88.86%. We consider the trained classification model as the simulated black-box.

In order to exemplify potential changes in the data distribution when applying the black-box to a different data domain, we introduce in the test set a new set of points, labelled as class 2, far from the class 2 original distribution. We can consider these points as a new cluster from class 2 that was not present in the source domain. Figure 3(d) shows the result of applying the trained classifier to the test dataset. Note that the new cluster falls in the class 1 region, inducing an erroneous prediction for all these points. Now the black-box classifier achieves a 71.76% accuracy score in the test data set.

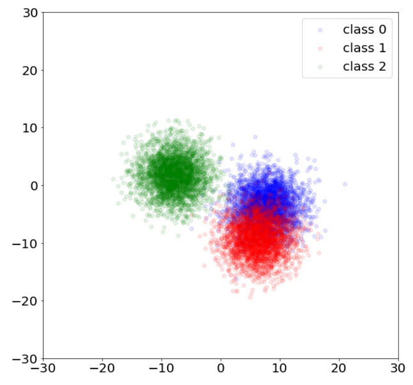
Observing the decision regions defined by the classifier, one can see three potential locations where predictions are prone to be wrong. The first error-prone region corresponds to the decision boundary between class 2 against the rest of the classes. The second one corresponds to the boundary between class 0 and class 1. Points close to these regions produce uncertain predictions. The third region of high uncertainty can be found in the area of the new class 2 cluster.

Figure 3 shows the results of applying the different methods for computing the uncertainty on this simple problem. Figure 3(b) shows the uncertainty measured by using the softmax-response of the predictions obtained with the black-box classifier. Figures 3(c) and 3(e) show the uncertainty obtained by computing the predictive entropy and variation-ratios respectively, by using a Gaussian random variables for the logits. Finally, Figure 3(f) displays the uncertainties obtained by using the proposed Dirichlet wrapper. Figures show the uncertainty metrics computed for each class, where the lighter the colour of the point is, the more confident the associated prediction. Looking at the results, we can see that the proposed method exhibits the best performance, detecting all enumerated sources of uncertainty.

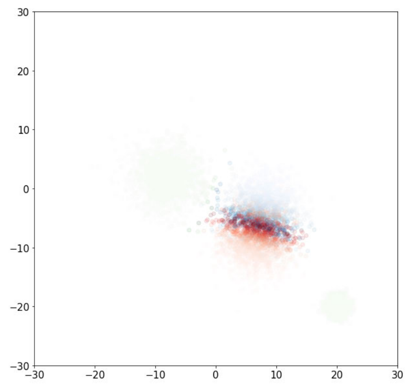
Looking in detail to Figure 3(b), we observe how this metric can detect the uncertainty from the decision boundaries, but it is not able to identify the new cluster. We have to recall that in this case, we consider the output of the classifier as a single point estimate, and we try to compute the confidence of the prediction based only on the output probabilities. We, therefore, apply a method based on the distance to the decision boundary of the predictions obtained, the softmax response, traditionally used as the confidence score for probabilistic classification systems.

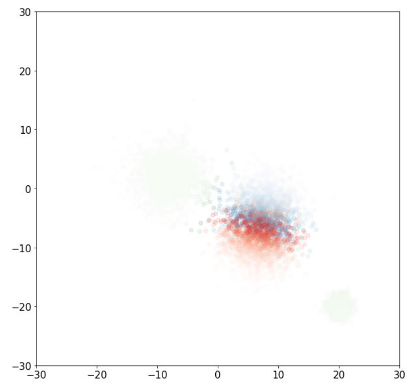
In figures 3(c) and 3(e), we apply a method similar to that proposed in [5] for capturing the aleatoric uncertainty. We would like to recall that in this case, we need to access the logits as they model the noise of the output predictions with Gaussian random variables with a diagonal variance term for each of the logits, u, as follows $u \sim \mathcal{N}(f^w(x^*), diag(\sigma^2(x^*)))$. This approach models uncertainty through the σ parameter expressing the variance of the distribution of the logits. As detailed in section III-C, after modelling the logits as Gaussian random variables, we still need to obtain a numerical score for the uncertainty. Hence, we use two different methods for obtaining such score: predictive entropy (Figure 3(c)) and variation ratios (Figure 3(e)). If we analyse the results obtained with the two uncertainty metrics, we see that each one focuses on a different type of uncertainty. Predictive entropy can capture the uncertainty of points near the decision boundary, similarly to softmax response. Variation ratios are able to capture the uncertainty associated with the noisy class two labels, while at the same time capturing some of the uncertain points at the decision boundaries.

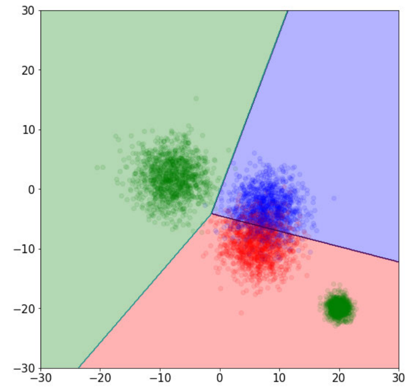
In order to understand this behaviour, we should examine the value of the standard deviation for each random variable.

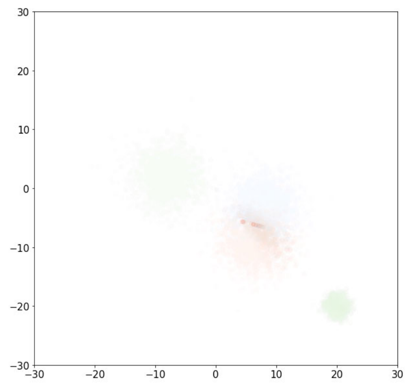


(a) Original synthetic dataset used to train the black-box classifier (BB)


(b) Uncertainty as measured using the softmax response of the BB


(c) Uncertainty as measured using predictive entropy using a Gaussian RV for the logits


(d) Learned BB applied to the test synthetic dataset


(e) Uncertainty as measured using variation ratios using a Gaussian RV for the logits

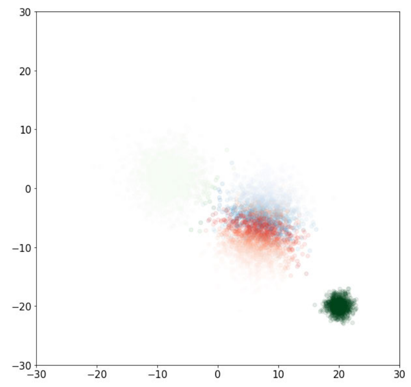

(f) Uncertainty as measured using predictive entropy resulting from the application of the proposed Dirichlet wrapper

FIGURE 3. Result of applying the black-box classification to the toy test dataset.

Observing the values for the σ parameter, we distinguish three different behaviours. First, for points that are far from the decision boundary between the original class 2 samples and the rest, the values for the sigma tend to be close to 0. As it gets closer to this boundary, the value of sigmas starts to grow until 20. Finally, the class 2 points that correspond to the new cluster at the bottom right, they all exhibit the higher σ values, from 20 to more than 40.

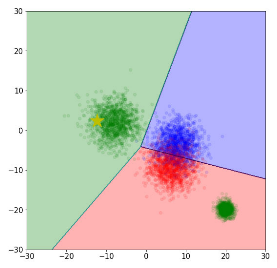
Based on these observations, we can conclude that the entropy of the original prediction heavily determines predictive entropy. As such, it can detect the uncertainty at the decision boundary. However, it is not able to identify the new class 2 cluster. All those points have a very confident (wrong) prediction, resulting in a minimal probability for the correct class. Then, even if the uncertainty associated with those points is high, the effect of the sampling on those points is diluted when averaging for computing the entropy value.

Regarding variation ratios, we observe more sensitivity for the different kind of uncertainties included in this toy problem. First, for points in the decision boundaries, subtle variations in their logit values likely forces a change in the predicted class for some cases. This accounts for larger uncertainty values associated with points close to the decision boundaries. More interesting is the new class 2 cluster. Changes in the predicted label have a significant impact in this case since we are merely counting changes and not averaging them as in the case of predictive entropy. Thus, even when the number of prediction changes is below 10%, the associated uncertainty is higher compared to those points that have no modifications.

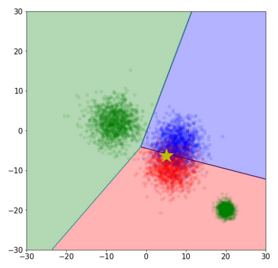
Finally, Figure 3(f) shows the uncertainty score obtained by using the proposed wrapper. Next subsection explicitly analyzes this figure and builds the intuition of the different types of uncertainty found.

A. ANALYSIS OF THE MEASURED UNCERTAINTIES WHEN USING THE WRAPPER

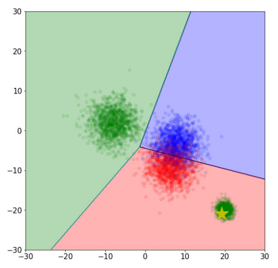
Now we analyse some examples to see how the proposed method works on different cases. To that effect, we will refer



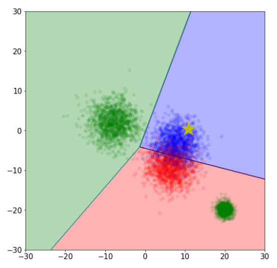
(a) Example of a point at very confident region from class 2



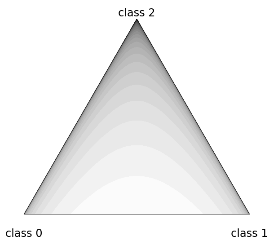
(b) Example of a point at the decision boundary between class 0 and 1



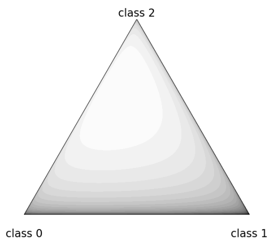
(c) Example of a point that belongs to class 2 but is predicted by the BB as class 1 with high confidence.



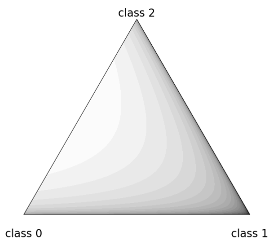
(d) Example of a point that belongs to class 1 but is in the class 0 region



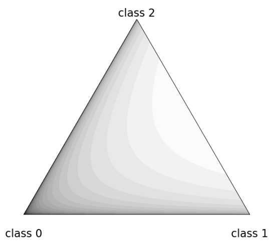
(e) We observe how, as the prediction is right, the concentration of the distribution is around class 2.



(f) We observe how the distribution concentrates in the line between the class 0 and 1.



(g) We observe how the distribution concentrates around class 1 even though the point belongs to class 2



(h) We observe how the distribution concentrates around class 0 even though the point belongs to class 1

FIGURE 4. **Examples of points from the toy problem together with the corresponding Dirichlet distribution.**

to Figure 4, where different cases are plotted jointly with their predicted Dirichlet distribution. The bottom line figures show the contour and concentration of the points distributed in the simplex of the three classes. Each class corresponds to one of the corners of the triangle: the bottom left corner one corresponds to class zero, the bottom right one to class one and the top corner to class two.

1) EXAMPLE OF A VERY CONFIDENT PREDICTION

First, we consider a point from class 2 that belongs to the original class 2 set at the left side of the figure. In this case, we would expect it to have low uncertainty, as it corresponds to a confident and right prediction. Figure 4(a) shows an example of this case as a yellow star. The point belongs to class 2, and the classifier outputs a prediction of $[3.01e-10, 2.75e-11, 1.00e+00]$, very confident of belonging to class 2. After applying the wrapper, we obtain a beta value of 0.11, quite high for the operational regime used. The alphas in this case are $[3.03e-11, 2.77e-12, 1.00e-01]$. Figure 4(e) shows the corresponding Dirichlet distribution. In this example, we see that the points concentrate around the top corner, that correspond to a very confident prediction of class 2, which aligns with their mean value $[2.54e-08, 2.54e-08, 9.99e-01]$. In this case, the entropy value is very low: $3.86e-10$.

2) EXAMPLE OF A PREDICTION AT THE DECISION BOUNDARY

Next, we examine a point from class 1 that corresponds to the set at the bottom of the figure, but close to the decision boundary between class 0 and 1. In this case, because the prediction is not that confident, we would expect it to have a higher uncertainty than before. The chosen point corresponds to the yellow star in Figure 4(b).

The point belongs to class 0, and the classifier outputs a prediction of $[4.12e-01, 5.87e-01, 2.67e-06]$, predicting the point as class 1 but with less confidence. After applying the wrapper, we obtain a beta value of 0.084, quite high for the operational regime used, but less than before. The alphas in this case are $[3.45e-02, 4.91e-0, 22.23e-07]$. As before, we observe the distribution in Figure 4 (f), where we see a higher variance. We can see that there is a higher probability concentrated around the line that connects class 0 and class 1. This is a clear indication that the confidence in that prediction is small. In this case, the entropy is a bit higher, 0.68.

So far, the uncertainty detected does not differ a lot from what a softmax response in a well-calibrated classifier can identify. The critical point of the proposed method lies in its ability to detect overconfident predictions; this is high confidence samples with wrong predictions.

3) EXAMPLE OF A PREDICTION FOR WRONGLY LABELLED POINTS

Consider now an example from class 2 in the new cluster. The chosen point corresponds to the yellow star in Figure 4(c). In this case, the point belongs to class 2, but the classifier outputs a prediction of $[1.13e-03, 9.98e-01, 8.86e-24]$, considering it to be class 1 with high confidence. Note that despite the high confidence value, the prediction is wrong if we are in the target domain. After applying the wrapper, we obtain a beta value of $1.06e-4$. The alphas, in this case, are $[1.21e-07, 1.06e-04, 9.46e-28]$ that produce the density displayed in Figure 4(g). In this case, the distribution shows large probabilities in the connections between class 1 and 0 and 2. Computing the expected value in this case, we obtain $[0.33, 0.33, 0.33]$ that corresponds to a value of entropy of 1.09. This assigns the point with considerable uncertainty.

4) EXAMPLE OF A NON-CAPTURED UNCERTAIN POINT

Finally, we would like to discuss a limitation of the proposed method for the toy scenario described here. Let us analyse, for example, a point that belongs to class 1 but is surrounded by class 0 points. The point corresponds to the yellow star in Figure 4(d). The point belongs to class 1, but the classifier outputs a prediction of $[9.93e-01, 6.60e-03, 3.07e-08]$ and considers it to be class 0 with high confidence. After applying the wrapper, we obtain a beta value of 0.087, not very small. The alphas, in this case, are $[8.73e-02, 5.80e-04, 2.70e-09]$ that produces the density distribution shown in Figure 4(h).

In this case, we observe that the higher density values are found around class 0 and 1. We can further see this effect by observing that the values corresponding to those classes are more significant in the expected value of the distribution, $[9.92e-01, 7.60e-03, 7.88e-10]$. In this case, the entropy is quite low, 0.044, and the wrapper fails to assign this point a substantial uncertainty value. The Dirichlet wrapper is not able to capture the wrong point, since, despite belonging to class 1, it is in a zone with a very high concentration of points labelled as 0.

5) EXAMPLE WRAP-UP

To sum up, we observe that for confident predictions, the value of the beta parameter is much higher than for those with high uncertainty. Here we would like to highlight that, in the proposed model, β belongs to the interval $[0, 1)$. This interval differs from other approaches that use Dirichlet for modelling the uncertainty, [9]–[11], where they work with unbounded values bigger than 1 for the β value. Working with values of $\beta > 1$ produces simpler uni-modal distributions. On the contrary, in the regime of work used in this work, the distribution is originally multi-modal, thus avoiding stability problems regarding the lack of control in the growth of the parameter.

Working in the interval $[0, 1)$, we observe that confident predictions usually take values of βs close to 0.1. When drawing samples from the corresponding Dirichlet distribution, this value alters a bit the expectation of the output, but still produces similar values as before. In the case of points where the black-box itself is uncertain (e.g. those close to the decision boundary) the wrapper assigns a lower β, around 0.05. This β still does not affect too much the generated prediction samples drawn from the Dirichlet. Thus, the final prediction keeps being uncertain. Finally, for those predictions that are very confident but correspond to very erroneous outcomes, the model estimates a very low β. In this case, the Dirichlet is forced to output points with high entropy, hence marking them as uncertain predictions.

Consequently, we claim that the proposed method seems better to capture the notions of uncertainty in this simple problem when working with the soft predictions of the black-box classification system. We now compare the results of applying the proposed uncertain parameter for the rejection system against the rest of the metrics analysed. Figure 5 shows the results of the three performance metrics described in section IV. From left to right, the chart displays the value of the corresponding metric after rejecting the percentage of points indicated. We observe how the softmax-response and Kendall's entropy exhibit similar results, as they struggle on detecting the noisy class 2 data points. The variation ratios obtained with the wrapper shows a bit better result than the previous two, but it is still insufficient to detect wrong predictions at low rejection ratios. The two best results correspond to Kendall's variation ratios and the predicted entropy obtained with the proposed wrapper. In the case of the later, by rejecting only 20% of the less confident points, we can increase the accuracy of the preserved data points in almost 20 points, outperforming the results obtained with Kendall's method. Besides, we want to highlight that in the case of the wrapper, the model has no access to the internal details of the black-box.

B. EVALUATING OUT-OF-DISTRIBUTION SAMPLES

Following the motivation of the Dirichlet wrapper and the role of uncertainty as an excellent proxy for capturing noisy labels when using black-box models in target applications, in this section, we motivate the addition of the out-of-distribution loss presented in section III.

Taking advantage of the toy problem introduced in the present section, we will show how to train the model to enable the detection of out-of-distribution data inputs. First, we generate random points drawing samples from a Uniform distribution centred around the original training distribution of the wrapper. The goal of adding these points is to simulate potential out-of-distribution inputs that might arise in prediction time. Next, we add these synthetic data points to the training dataset defined as in figure 6 (a). To differentiate between the two types of points, we add an extra label with values $y_o = 1$ for the original data points and $y_o = 0$ for the newly generated data inputs. With the augmented dataset, we trained the wrapper model using the out-of-distribution loss function as defined in equation 9.

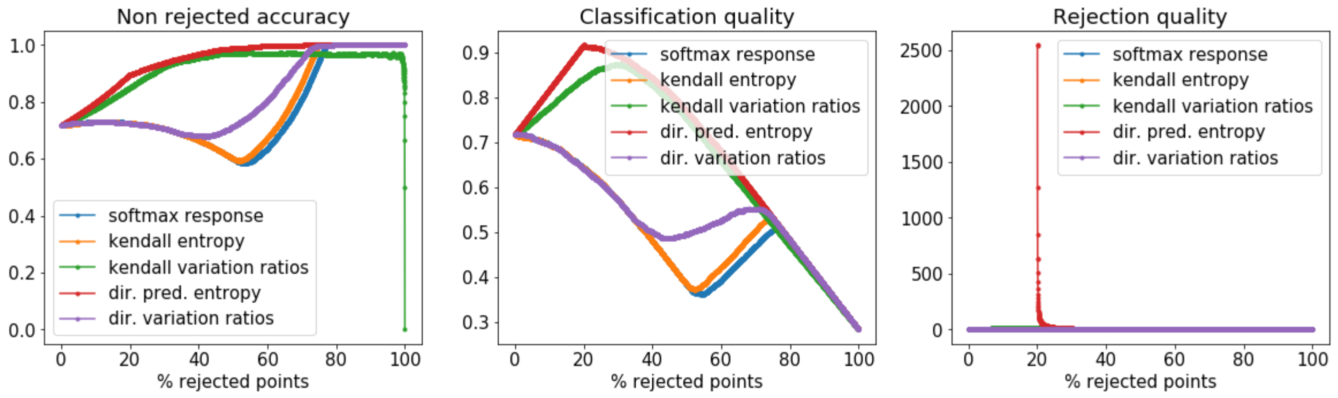


FIGURE 5. Classifier with rejection using different uncertainty metrics: softmax response, Kendall's predictive entropy and variation ratios and Dirichlet wrapper predictive entropy.

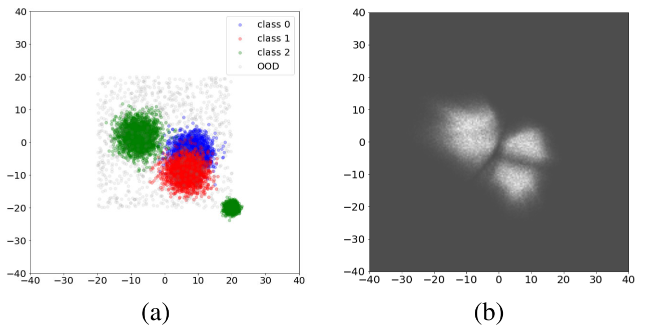


FIGURE 6. (a) Training set including out-of-distribution points from $U[-20, 20]$ and (b) Uncertainty of points in $U[-40, 40]$ (The darker, the more uncertain).

Figure 6(b) shows the results of applying the out-of-distribution wrapper, the darker, the more uncertain the prediction is. In order to validate the proposal, we further draw the samples from a broader Uniform distribution, corresponding to a 100×100 bounding box around the test points, instead of the 20×20 one used for training. In this respect, we validate that the method generalizes for out-of-distribution samples not seen during the training stage. The results show that the out-of-distribution wrapper preserves the ability to detect uncertain predictions, showing similar results to those obtained in 3. Moreover, we also note how the model identifies the new out-of-distribution points by assigning a high uncertainty to those points.

To sum up, we conclude that the out-of-distribution wrapper here proposed is a handy tool for assessing the performance of a black-box classifier given a new target application. With only a small fraction of labelled data from the target domain, we can train the model to estimate the uncertainty of future predictions. This uncertainty can help with the evaluation of different black-box APIs or can provide an actionable mechanism to improve the performance of them in production through rejection strategies. We also show how by augmenting the target datasets and using a new loss function, we can deliver a more robust method of uncertainty estimation that can prevent the model from degrading in case of future changes in the distribution of the inputs.

The experiment described in the present section validates the proposed method as illustrated with a toy scenario. In the next section, we showcase how the wrapper is also capable of modelling uncertainty for more complex situations dealing with real data.

VI. EXPERIMENTAL SETUP

This section contains the experiments carried out to validate the effectiveness of the proposed model using real data. The tests included aim at simulating the central use case of this work, i.e. to evaluate the performance of a pre-trained black-box in a target scenario, and how to respond when this performance is not satisfactory.

In all the cases, we take a source domain with a large number of labelled training samples and proceed to simulate a black-box classification model for this source domain. Next, we freeze this model and pretend its exposition as it was an API part of an MLaaS platform.

We then change roles and act as a consumer of this API. At this stage, we assume that we do not have any information about the details of the API and we want to evaluate it when applied to a given target domain with a small number of labelled training samples. The first thing to do in this case is to assess whether or not the output of the API, the predicted categories, fit our problem or whether a mapping function to adapt the API results to the target use case is required. Then, we proceed to call the API for our target dataset and obtain the output predictions, calling the mapping function when needed. In order to simulate the case when the classifier outputs hard predictions, we generate a new set of predictions based on the probabilities obtained with the black-box. However, only the class with maximum probability is reported as output.

Next, we take the target training samples together with the predictions obtained to train the Dirichlet wrapper, after what we can call it to estimate the uncertainty score associated with each prediction. At this point, we can use the score to reject uncertain predictions or to evaluate the performance on each class and decide whether to keep using the API, try another one or collect more data to train our model.

Scenarios proposed to validate the method include use cases in Natural Language Processing and Computer Vision. In particular, we consider the problems of sentiment analysis and image classification. In both cases, pre-trained models are applied to solve different tasks than those for which they were trained. This scenario shows how the use of uncertainty to detect problematic examples can help in increasing the overall quality of the predictions obtained for the new task.

Next subsections, validates the method in different use cases and shows how to use it for rejecting erroneous predictions according to the predicted uncertainty.

A. A NATURAL LANGUAGE PROCESSING USE CASE

The impact of domain shift and the consequent domain adaption of machine learning models is very relevant in the field of natural language processing. Many previous works have studied this issue in NLP tasks in general [33] and sentiment analysis in particular [34], [35]. The central goal of these contributions was to train models that are capable enough to capture the differences between domains and be able to generalize across them. Differently from these words, our proposal focuses on the problem of detecting this shift between the training dataset of an original black-box model and the data of a given application and provide a means to identify difficult examples that may lead to overconfident and wrong predictions.

In order to validate the method proposed, we applied the wrapper to an NLP-based sentiment analysis system applied to product reviews. The experiment consists of training a sentiment analysis classifier using reviews of a given domain and evaluating the uncertainty of the predictions obtained when applying the pre-trained model to a new domain.

To illustrate real scenarios, we include two types of experiments: in the first one, we train the black-box with Amazon products of a section with multiple reviews. These reviews correspond to 1 to 5 stars assessments made by Amazon's users. Then we apply the black-box to reviews of another department and evaluate the performance of the model. The second experiment consists of training a model using reviews of Yelp venues, rated as 1 to 5 stars, and apply the model for IMDB movie reviews in a binary setting. In this case, we need to provide the mapping function that transforms the original five classes to a binary problem. Finally, for the sake of completeness, we also apply the Yelp classifier to the Amazon's product reviews and the Amazon classifier to the SST-2, adapting in each case for the number of target classes. In total it adds to 30 experiments.

The experiments are conducted using the following datasets:

- Amazon Multi-Domain Sentiment dataset contains product reviews taken from Amazon.com from many product types (domains) [36]. The dataset consists of ratings from 1 to 5 stars despite 3-star reviews, i.e. reviews with neutral sentiment were not included in the original.
- Yelp challenge 2013,[5] the goal is to classify reviews about Yelp venues where their users rated them using 1 to 5 stars. This dataset is used to train the black-box applied to the SST-2 problem. As mentioned, we need to transform the Yelp dataset from a multiclass set to a binary problem, grouping the ratings below three as a negative review, and as positive otherwise.
- Stanford Sentiment Treebank [37], SST-2, binary version where the task is to classify a movie review is positive or negative.

TABLE 1. Datasets used for NLP experiments.

	Train	Validation	Test	N. classes
Amazon apparel	5,828	648	2,776	4
Amazon camera	4,666	519	2,223	4
Amazon computer	1,994	222	555	4
Amazon dvd	89,593	9,955	24,884	4
Amazon electronics	14,495	1,616	6,903	4
Amazon health care	4,551	506	2,168	4
Amazon kitchen	12,509	1,390	5,957	4
Amazon magazines	2,639	294	1,258	4
Amazon music	109,733	12,193	52,254	4
Amazon toys	9,465	1,032	2,630	4
Amazon videos	22,793	2,533	10,854	4
SST-2	65,538	872	1,821	2
YELP2013	186,189	20,691	22,991	5

Table 1 lists the datasets used for the NLP experiments.

We train a deep learning classifier for the big datasets(Yelp2013 and Amazon's music and DVD product reviews) using a model based on a representation obtained averaging the Word2vec embeddings [38] on the review words. Once trained, we use the model to predict the sentiment using another the rest of the datasets.

After obtaining the predictions for each target domain, we used the proposed Dirichlet wrapper to estimate the uncertainty score for each prediction. Finally, we evaluate the accuracy of the predictions using different values of the uncertainty score for rejecting uncertain examples.

B. IMAGE CLASSIFICATION

Similar to the NLP case, in computer vision applications, the adaptation of the trained models to new domains is essential. In the literature, one can find different works that tackle the problem using different approaches [39], [40], including the observation of discrepancies in the distributions of both source and target domains or proposing a conditional adaptation network for cross-domain image classification. In our work, we focus on estimating the uncertainty of every single prediction by applying the proposed wrapper and use this uncertainty to discard overconfident and erroneous predictions.

[5]https://www.yelp.com/dataset/challenge

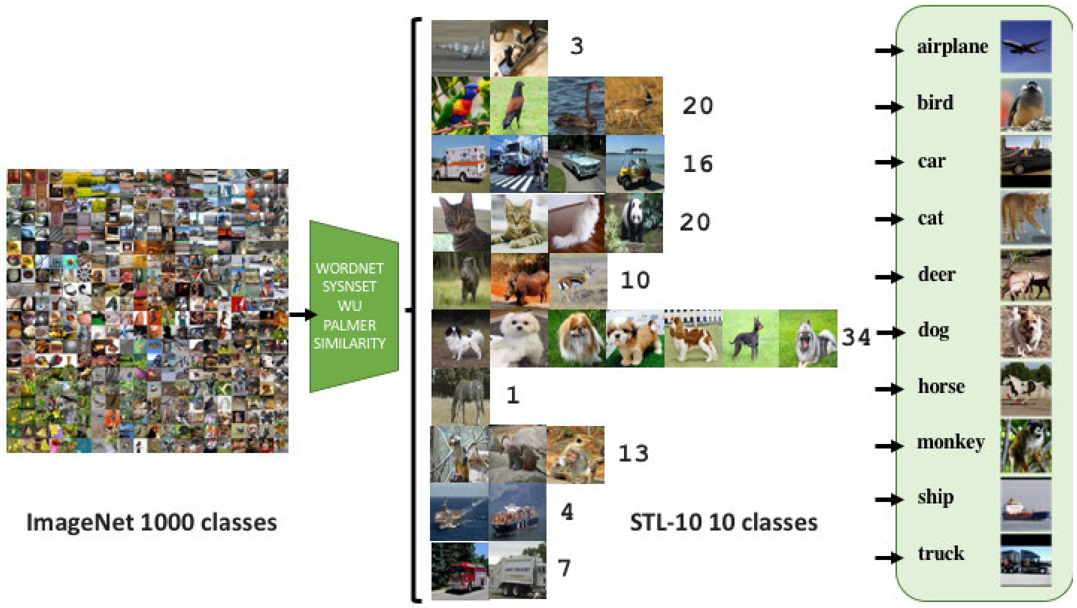


FIGURE 7. Mapping from the 1000 classes of ImageNet to the 10 of STL-10 based on the Wu-Plamer similarity of the ImageNet sysnet as definied in WordNet to the corresponding STL-10 cateogry.

In this use case, we show how the proposed wrapper applies in transfer learning scenarios. Here the simulated API, the reference pre-trained model, are different Keras[6] implementations of a MobileNet.V2 [41], DenseNet [42] and Inception.V3 [43] models, trained for classifying 1000 classes of the ImageNet dataset [44].

As the target domain we use the STL-10 dataset [45]. Due to the difference in the output categories, a mapping between both domains is needed. Figure 7 illustrates the mapping process used to adapt the ImageNet pre-trained models to the 10 STL-10 classes. Taking advantage of the fact that each ImageNet label belongs to a SynSet in WordNet, we assign a SynSet for each of the 10 STL labels. Next, we use the Wu-Palmer Similarity [46] to find those ImageNet Sysnet that are more similar to the chosen STL ones.

Thanks to this mapping method now we can assign ImageNet predictions to the STL-10 labels. For each new example, we use the ImageNet models to obtain the probabilities for each of the 1000 categories. Then we use the mapping to get only the probabilities of the categories mapped with the corresponding STL-10 classes. For each STL-10, we compute the weighted average, considering the number of assigned ImageNet classes for each STL-10 label. In addition to the labels mapping, we needed to rescale the size of STL-10 images, which were $96 \times 96 \times 3$, to the ImageNet size, $224 \times 224 \times 3$. Later on, we use these images together with the corresponding predictions as the input for the Dirichlet method to estimate the uncertainties.

Finally, we proceed in the same way as done in the former case, using the uncertainty score to evaluate the performance of different rejection points.

[6]https://keras.io/applications/

VII. RESULTS

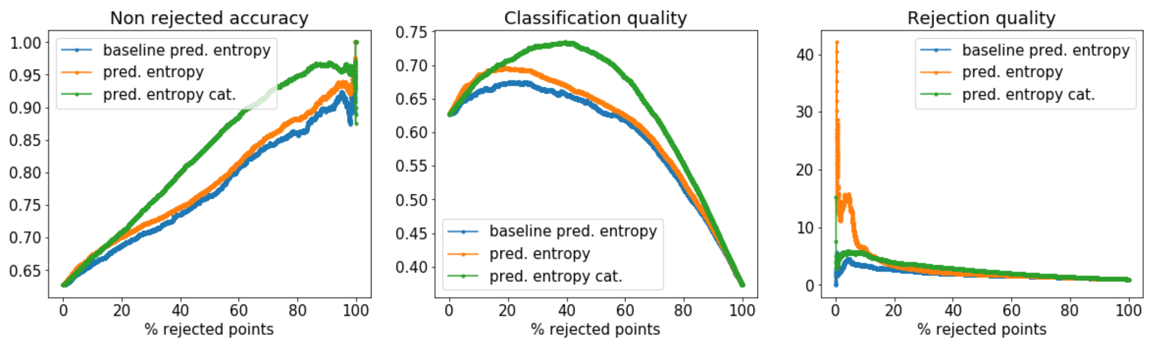
This section presents the results obtained in the different experiments carried out for both the NLP and CV domains.

A. RESULTS IN THE NLP DOMAIN

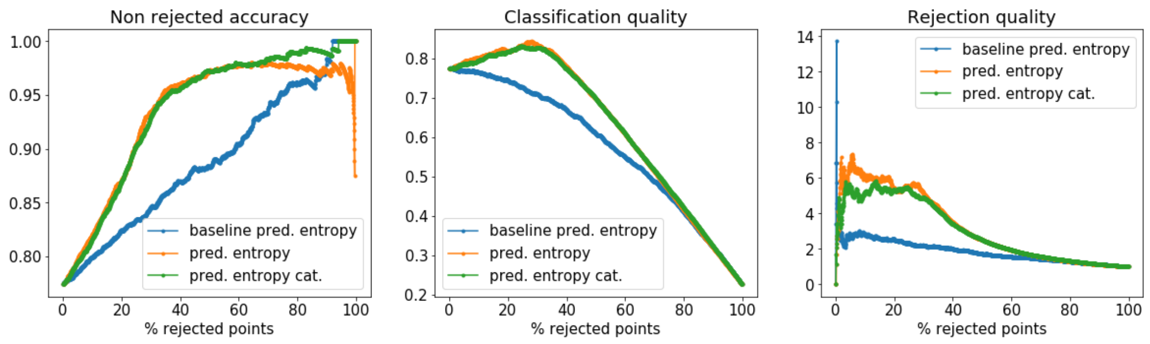
Figure 8 compares the performance metrics of the three rejection measures: the entropy of the predictions of the black-box (*baseline*), the predictive entropy of the wrapper (*pred. entropy*) and the predictive entropy of the wrapper when applied to a categorical output of the black-box (*pred. entropy cat.*).

First, we compare the two uncertainty metrics derived from the wrapper when applied to soft and hard predictions with the predictive entropy of the black-box. This metric is obtained directly from the predictions of the pre-trained model by computing the entropy over the softmax probabilities. It is worth noting that one can replace predictive entropy by other different metrics without differences in the results (see Appendix A). Moreover, we would like to remark that the comparison with the predictive entropy obtained for categorical predictions is included as a reference, taking advantage of the fact that here we simulate the API. In the real world, the baseline used here will not be available as it is based on the output probability distribution.

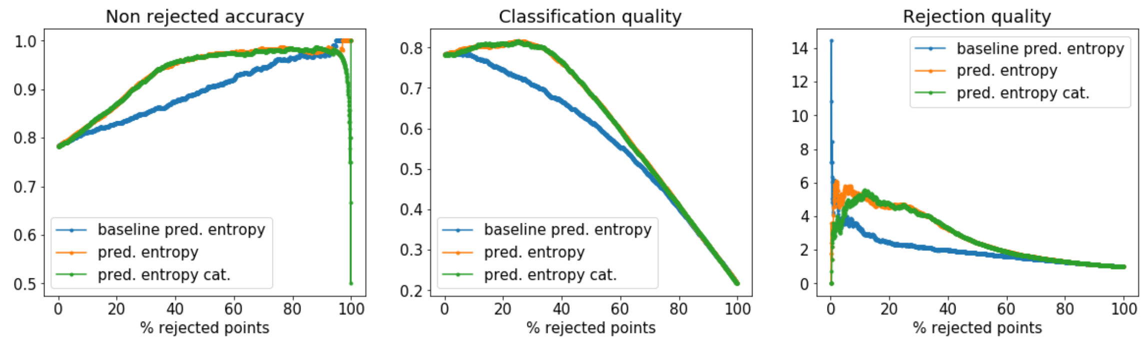
On each experiment, the three pictures refer to the three performance metrics (non-rejected accuracy, classification and rejection quality) on each rejection point. The X-axis corresponds to the percentage of points rejected at a given point, and the Y-axis depends on each performance metric. For the non-rejected accuracy, the value displays the accuracy of the predictions for the non-rejected points. For the classification quality, the value shows the percentage of points that



(a)



(b)



(c)

FIGURE 8. **Apply Amazon Music BB to Video reviews, Yelp 2013 BB to SST-2 movie reviews and Amazon video BB to SST-2 movie reviews.**

are right and not rejected plus the wrong and rejected versus all the points. Finally, the rejection quality corresponds to a proportion between the rejection and the misclassification. In the first and second metrics, the closer the value to 1 the better, while in the third one values above 1 mean a good rejection point. An optimal rejection point, therefore, will be one that combines excellent performance in the three metrics.

To apply the rejection, we proceed as follows. For each point in the test set, we proceed to apply the uncertainty function. In the case of the black-box predictive entropy, the value is obtained directly from the prediction, computing the entropy of the probability distribution of the softmax. For the Dirichlet predictive entropy and the variation ratios, we apply the wrapper to the input and the corresponding black-box prediction to obtain the uncertainty. Once we have a value of the uncertainty for each point, we sort from more to less uncertain, and we start to reject those more uncertain. Each point in the X-axis shows the performance of the resulting classifier after rejecting $x\%$ points in the test dataset. Each line in the plots displays the performance of each of the uncertainty functions compared.

According to the results obtained, the predictive entropy obtained after the proposed method shows better behaviour in all scenarios and metrics. As we remove more samples according to the uncertainty, the proposed method displays much better accuracy than its counterparts. These results validate the hypothesis that the aleatoric uncertainty computed by the wrapper effectively captures the confidence in the prediction and the samples prone to error. Note that, although our proposal performs much better, its absolute gain depends on the scenario. In domains where there is a need to adapt the original model to the particular target application, there is more to gain by using the wrapper. If we observe the classification quality (plot at the centre of each figure) and the rejection quality, we can see that the proposed metric is also excellent at rejecting the miss-classified points.

Concerning the predictive entropy for the categorical version of the black-boxes, we observe that when compared with

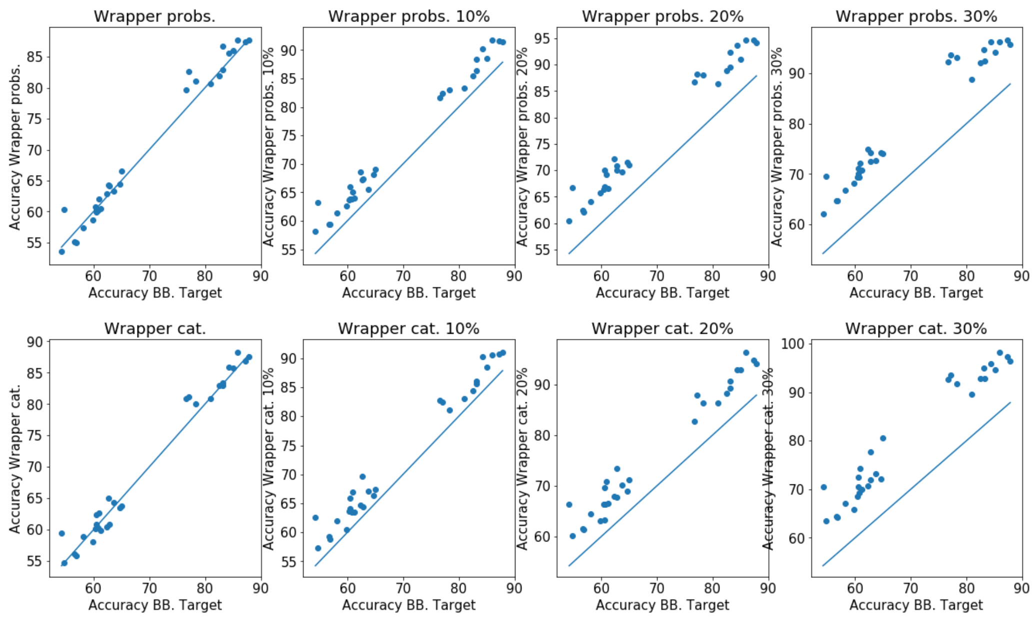


FIGURE 9. Non-rejected accuracy of the wrapper compared to original BB with both probabilistic (a) and categorical (b) outputs.

the performance of the baseline and the predictive entropy of the soft predictions, in some scenarios we can observe that the performance is a bit lower. However, we have to recall that this comparison is not fair as in categorical scenarios, we will not have access to the probabilities. Even though we still appreciate that it is a good rejector, even when the predictions are categorical.

Figure 9 displays a summary of the results obtained. On each plot, the x-axis corresponds to the accuracy of the black-box as applied to a target domain.[7] Y-axis shows the accuracy of the predictions obtained by applying the wrapper model, and the non-rejected accuracy for 10%, 20%, and 30% of the points using the predictive entropy. We show the results for both the probabilistic and categorical versions of the black-boxes. We also plot a line that displays the accuracy of the BB to compare the results.

In all the experiments, the accuracy obtained when using rejection outperforms the original black-box, and we can observe how by increasing the number of rejected points, the accuracy increases accordingly. Even though the gains are higher for the probabilistic version, we still can appreciate how the predictive entropy for the categorical predictions behaves like a good rejector too.

Moreover, we would like to remark the value in the first plot, that displays the accuracy of the wrapper applied to the target data source. In this case, to predict the values, we follow the same approach as for the entropy: once we have modelled the output as a Dirichlet distribution, we sample from this distribution N times and take the expected value as the prediction. In this case, we would like to highlight how just applying the wrapper on top of the original black-box, in some applications, we see an increase in the accuracy without the need of rejecting any sample.

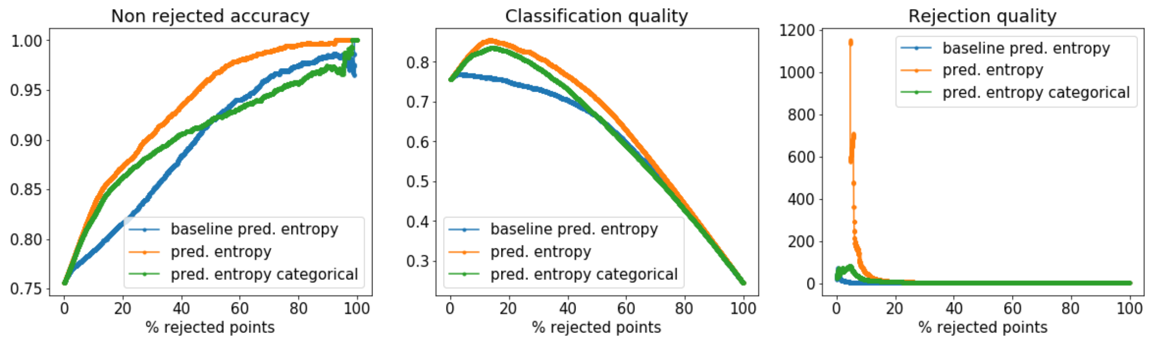
Table 3 and Table 4 shows a detail of the numerical results obtained during the experiments for the different combinations tested, both for probabilistic and categorical outputs. The first column, black-box source acc, describes the accuracy obtained for the source dataset after training the original classifier. Next, column black-box target acc. describes the accuracy obtained when applying the black-box to the target dataset. The third column displays the result of obtaining predictions by applying the wrapper model, sampling from the learned Dirichlet output distribution. The rest of the columns show the non-rejected accuracy and the classification and rejection quality after rejecting 10, 20 and 30% of the points, using the proposed predictive entropy as a rejector.

B. RESULTS IN THE COMPUTER VISION DOMAIN

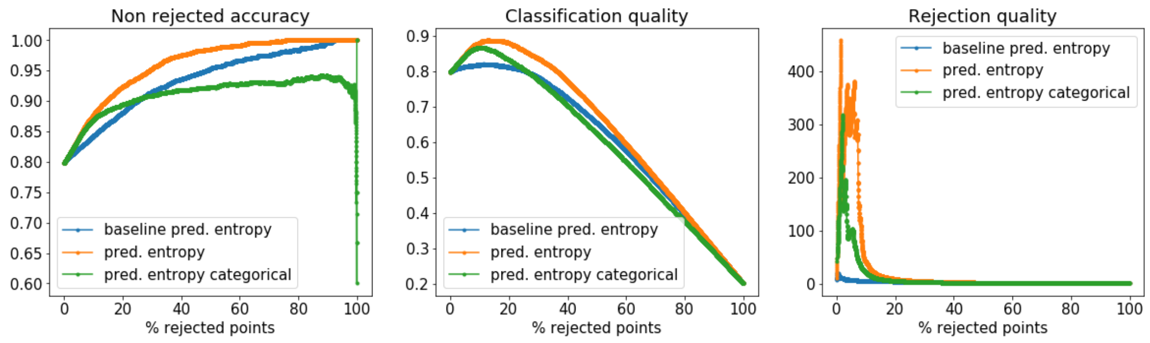
Similarly to the NLP experiments, Figure 10 displays the performance metrics of the three rejection measures (*baseline*), *pred. entropy* and *pred. entropy cat.*) for the tests run in the computer vision case.

In this domain, we can see that by only rejecting 10% of the more uncertain predictions, we increase in 8 points the accuracy for the remaining images. This increase goes over 12 points when rejecting 20% of the less confident predictions. In all the cases, the results using the predictive entropy of the wrapper outperform the baseline of using the softmax response of the BB. Observing the values for the classification quality, we can determine that it takes its maximum when

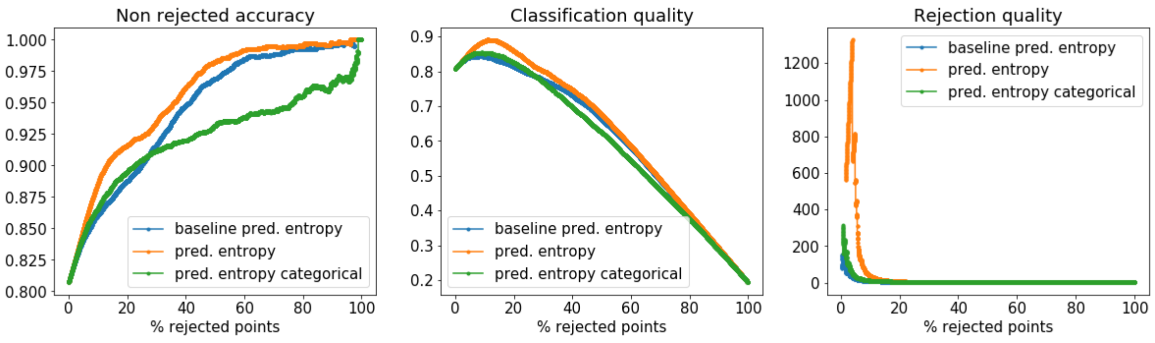
[7]The gap between accuracy scores correspond to the difference of results obtained from the binary and multi-label classifiers applied for movies and Amazon products respectively



(a)



(b)



(c)

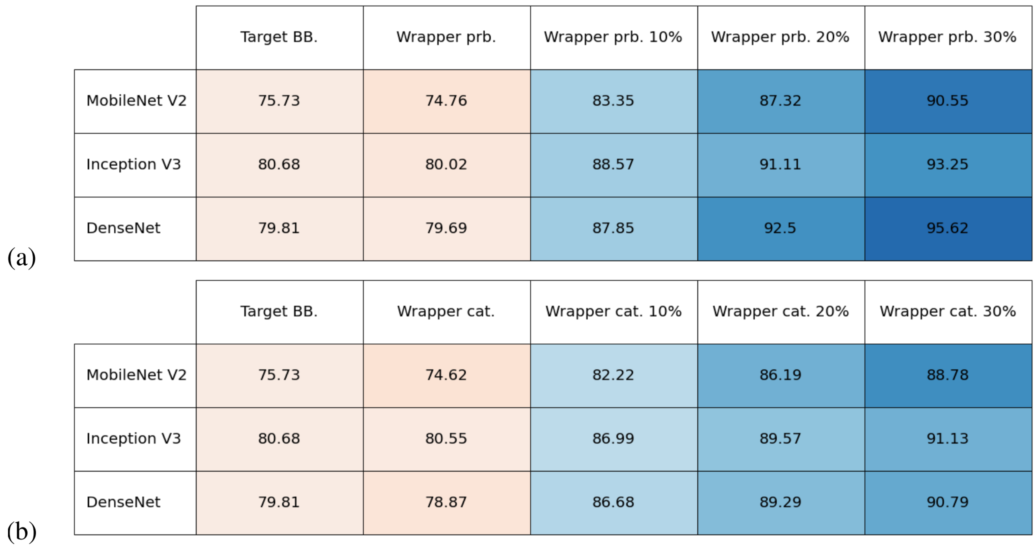
FIGURE 10. Apply ImageNet BB trained with MobileNet V2 model to STL-10, Apply ImageNet BB trained with DenseNet model to STL-10, Apply ImageNet BB trained with Inception V3 model to STL-10.

discarding 20%, indicating that this could be a good rejection point for these cases.

About the categorical version, we observe that it behaves differently depending on the original black-box model. For MobileNet.V2, we see that it outperforms the baseline for the first 40% predictions. For the rest of the models, we observe that this threshold falls to around 30%. This difference might be explained by the fact that during the selection of the model used for implementing the wrapper, we observe that MobileNet showed the best results. However, as commented for the NLP case, uncertainty still appears as a good rejector, especially if we bear in mind the fact that there is no chance to use the softmax response or similar metrics in such scenarios.

Figure 11 illustrates the overall results obtained in the computer vision scenario for both probabilistic and categorical versions of the black-box. In this domain, even though the results of rejection are outstanding, we observe how the accuracy of the prediction obtained from sampling the output from the wrapper, with no rejection, does not improve the results obtained from applying the original black-box. This lack of improvement might be caused by the fact that for this domain, and due to the need for mapping the BB and target labels, the cases of over-confident predictions might be very relevant. For preventing this overconfidence, rejection is a handy tool, as the results show.

After analysing the results obtained, now it is time to design the rejection mechanism for the classification model. Take, for example, the use case where we are using the ImageNet model to predict STL-10 images. Looking at the values of the classification and rejection quality measures, they show their maximum around 10% of rejected points, (84.67% and 54.32% respectively). Observing the accuracy of the kept examples, 83.54%, we conclude that only by rejecting this 10% we increase the accuracy of the model in almost 8 points.



(a)

	Target BB.	Wrapper prb.	Wrapper prb. 10%	Wrapper prb. 20%	Wrapper prb. 30%
MobileNet V2	75.73	74.76	83.35	87.32	90.55
Inception V3	80.68	80.02	88.57	91.11	93.25
DenseNet	79.81	79.69	87.85	92.5	95.62

(b)

	Target BB.	Wrapper cat.	Wrapper cat. 10%	Wrapper cat. 20%	Wrapper cat. 30%
MobileNet V2	75.73	74.62	82.22	86.19	88.78
Inception V3	80.68	80.55	86.99	89.57	91.13
DenseNet	79.81	78.87	86.68	89.29	90.79

FIGURE 11. Results of applying the wrapper to the target STl-10 dataset with different rejection points.

If we examine the value of the uncertainty score predicted at the selected rejection point, 2.3024, we can see that this value is quite close to the maximum predictive entropy for ten classes, 2.3026. Thus we can see that the proposed wrapper is taking the more uncertain and erroneous points to the maximum entropy, as we saw in the toy scenario for the wrong labelled class 2 points. Filtering the predictions with uncertainty scores higher than this 2.3024 value, we discarded 798 of the 8000 test data points.

In case we wanted to put this system in production for classifying new STL-10-like images, for those predictions that reached an uncertainty score higher than 2.3024, we could warn the user about the uncertainty of the prediction, or directly discard those examples.

To find the right balance between the number of rejected points and the accuracy of the kept data will depend on each use case, as in some cases we would rather high accuracy no matter the number of discarded points. In contrast, in some other circumstances, the system will have to give a prediction for all the data points. In any case, the proposed method will always be able to indicate a level of confidence for the prediction, leaving it in the hands of the user to make the final decision.

VIII. DISCUSSION

A. UNCERTAINTY EVALUATION ON UNKNOWN CLASSES

During the description of the image classification use case, we mention the necessity of developing a mapping between the 1000 ImageNet labels and the 10 used in STL. By using similarities between the WordNet SynSets that defined the ImageNet categories and those selected for STL, we created a correspondence between each STL category and N matching ImageNet labels.

Although in many cases the mapping can identify clear matchings for each STL-10 category in those predicted in ImageNet, there are some cases where the equivalence is not that evident. In Figure 7, we see the number of ImageNet categories that are bound to each STL-10 category. One can see that there are categories well represented, dogs mapped to Japanese spaniel, Maltese dog, Pekinese, Shih-Tzu, up to 34 different categories; others like horses have an inferior representation. In the latter case, the only mapping identified with horses is the zebras. Beyond the biological discussion of their similarities, it is clear that in this case, the level of uncertainty might be high when applying the ImageNet to identify horses, as it shows when looking at the distribution of the uncertainties obtained for the "horses" category displayed in Figure 12(a).

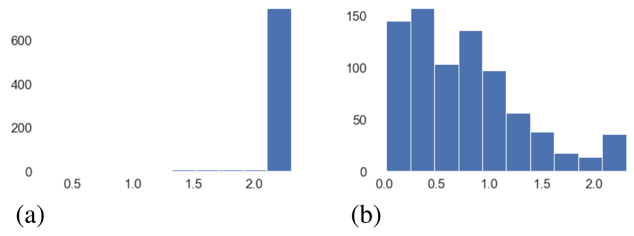


FIGURE 12. Distribution of uncertainty scores for the "horses", (a), and "deers" (b) categories in STL-10.

Moreover, in classes that exhibit more matches, some of those matches are not very accurate. Take, for example, the deers category. In the WordNet mapping, we find similarities with hartebeests, impalas, gazelles, but also with wild boars, warthogs, hippopotamus, water buffalos, bison, Arabian camels or llamas. Figure 12 (b) shows the distribution of uncertainties for this class. In this case, we see that the model is more confident than in the case of horses. We conjecture that this may be because we consider the weighted contributions of the mappings, and this is helping to identify deers as a combination of those animals.

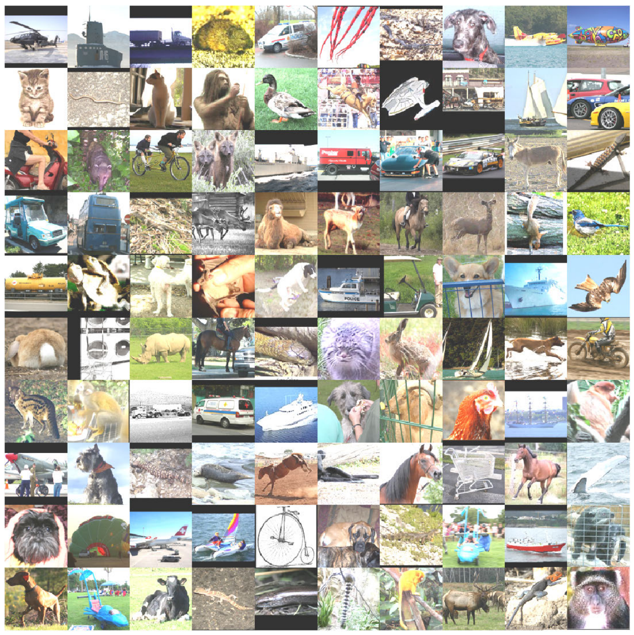


FIGURE 13. Examples of images included in the unlabeled set of STL-10 images.

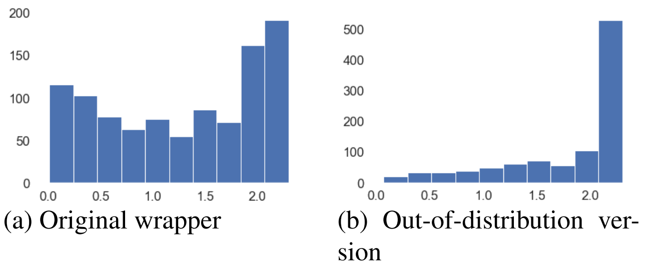


FIGURE 14. Histogram of the distribution of uncertainties of the test unlabeled images from STL-10 obtained using the two wrapper versions.

B. OUT OF DISTRIBUTION IMAGE CASE

Beyond the uncertainties induced by the possible mismatches in the label mapping, in this section, we also analyse the role of the out-of-distribution term added to the loss function of the wrapper. To validate the out-of-distribution method, we take advantage of the fact that the STL-10 data set incorporates, according to the description, "100000 unlabeled images for unsupervised learning". These examples are extracted from a similar but broader image domain. Figure 13 show some examples of these images that contain other types of animals (bears, rabbits, etc.) and vehicles (trains, buses, etc.) in addition to the ones in the labelled set.

In training time, we added 1000 of these unlabeled images to the STL-10 training set for learning the uncertainties, labelling them according to whether they belong to the distribution or not. After training the out-of-distribution wrapper, we take 1000 images more from the unlabeled data set to test the method.

In figure 14, we can see the results of applying the original wrapper and the new out-of-distribution version to the test unlabelled images. We can observe how the new version outperforms the previous one detecting the out-of-distribution images by assigning a high entropy to them. Thus, on prediction time, the out-of-distribution version of the wrapper can detect not only uncertainties associated with noisy labels of images included in the training set but also can protect against the addition of images that are away from the original distribution. By analysing the uncertainty of the predictions, the practitioner will be able to identify such new categories and decide on whether discard them or train a specific model for them.

Moreover, in figure 15, we analyse the performance of the new method compared to the original one with regards to the rejection ability when applied to the labelled test images. In this part of the experiment, we use both the original wrapper and the out-of-distribution version to the test set in STL-10 to compute the prediction uncertainty. Next, we use these uncertainties for rejecting uncertain predictions, as in section VI-B. We observe how, despite a small reduction of the rejection performance, the wrapper still retains its ability to detect the uncertainty in erroneous predictions, outperforming the softmax response.

C. RELATION WITH CALIBRATION METHODS

As mentioned in the related work section, calibration methods ensure that the output of probabilistic classification systems describes a proper probability distribution. Thus, one can take advantage of these probabilities to not only emit a prediction but also to estimate the confidence of such prediction. Calibration methods, as described in [47], are similar to the proposed uncertainty wrapper in the sense that they actuate on already trained models, working on the output of the classifier to deliver a measure of confidence. Nevertheless, some differences make the proposed method to be more generic and applicable.

First of all, in the present work, we proposed a method for evaluating the predictions obtained using a pre-trained black-box system, checking whether the obtained results fit for the current problem or not. It is not the goal of this method to alter the predictions obtained, but to evaluate their appropriateness for the given domain. Calibration, on the other hand, alters the probabilities of the output. This alteration, though, will not affect the results when using those predictions for rejecting unconfident points, as it is the order of this probabilities what matters.

However, the main difference comes from the constraint imposed by black-box systems: we do not have access to the internals of the model. The calibration methods described in [47] all work at the logits level,[8] which prevent them from being applied in the scenario here described. Moreover, as outlined in the results, the proposed method also applies when the output of the black-box system is a hard categorical result and not a probabilities distribution, situation in which the use of calibration is not possible.

[8]Logit activations are those previous to the last activation function, usually a softmax function in neural networks applied to classification.

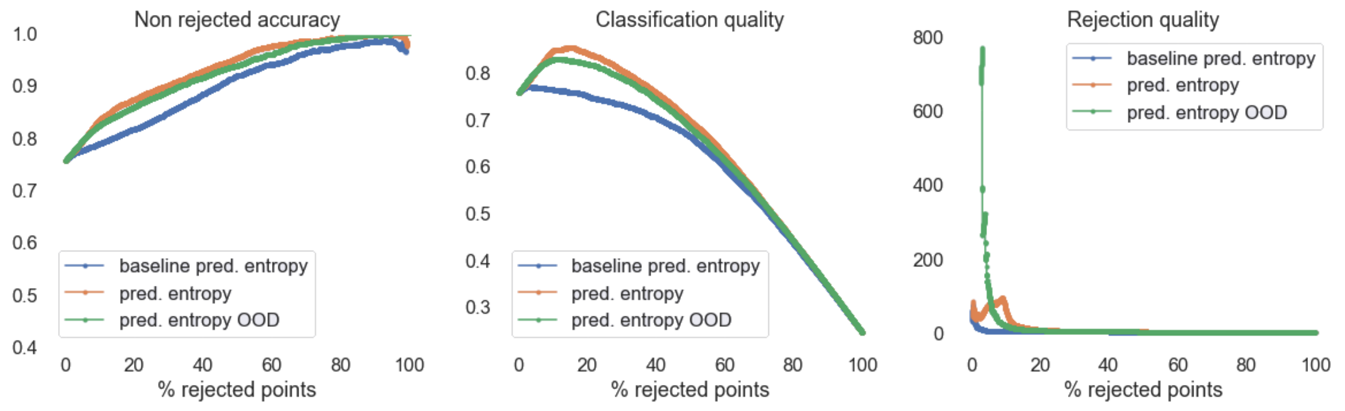


FIGURE 15. Rejection performance metrics comparing the results from applying the softmax response, the predictive entropy of the original wrapper and the OOD version.

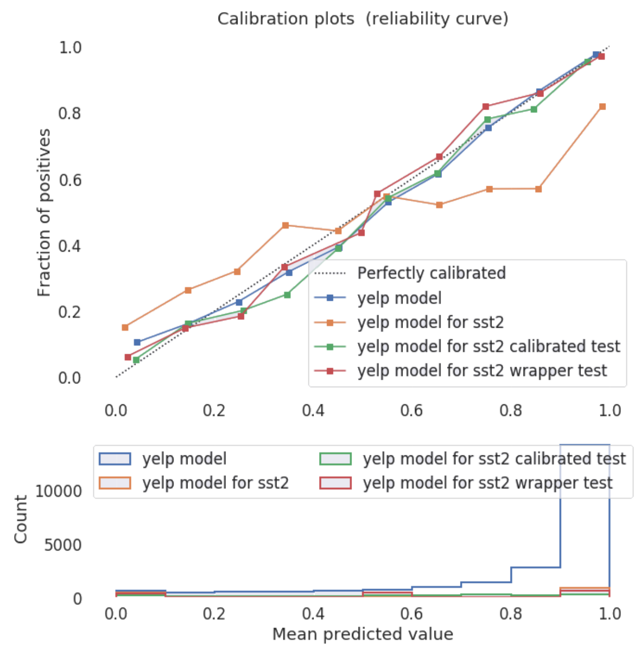


FIGURE 16. Reliability diagram displaying the calibration of the black-box applied to the source, target domain and after calibrating for the target domain, compared to the calibration obtained by the predictions from the wrapper.

To illustrate the relationship between calibration and the proposed uncertainty wrapper, we take advantage of the simulation of a black-box scenario in the NLP test cases. There, we have access to the logits of the original model, so we can apply a calibration method and observe how it relates to the wrapper.

In the case of training a reviews classifier using Yelp reviews and apply it to classify movie reviews from the SST-2 data set as positive or negative, the first thing that we observe in figure 16 is that the model trained with the source dataset is well calibrated. We also notice that when applied to the target domain, the calibration is lost, as expected, as the distributions of both datasets are different. The most relevant result observed in this plot is the fact that the result after applying a recalibration method, Temperature scaling [47], to the SST-2 predictions are similar to the calibration of the probabilities obtained directly from the wrapper, so the proposed method already yields calibrated predictions.

Table 2 shows the Expected Calibration Error, ECE, [48], computed for each of the experiments included in the present paper in the following scenarios:

- ECE of the Black-box applied to the original data source, except for the case of image classification, where we do not have access to the initial training set.
- ECE of the Black-box applied to the target data source.
- ECE of a calibrated version of the Black-box using the Temperature scaling method applied to the target data set.
- ECE of the predictions obtained from the wrapper applied to the target data source by drawing 1000 samples from each Dirichlet distribution.

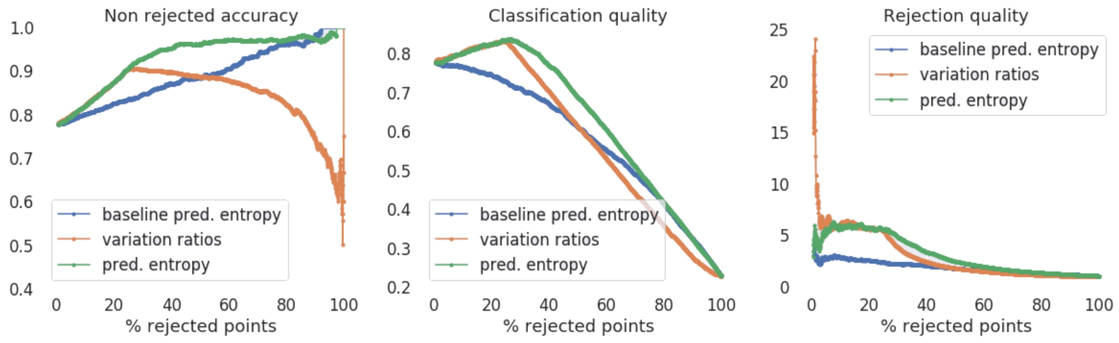
Results show that predictions obtained using the wrapper are similar to those obtained after applying the calibration or even better, as in the case of image classification.

Next, we validate that the calibration has no direct impact on the outcomes of the wrapper. We use temperature scaling [47] for calibrating the predictions for the target dataset. We then run two different experiments to evaluate the impact of calibration in the proposed method:

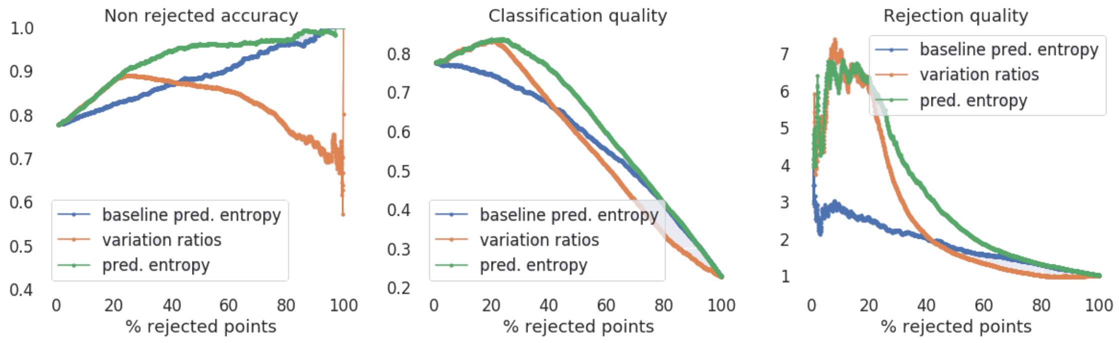
- Compare the wrapper uncertainty with the predictive entropy of the calibrated output. In this experiment, we validate that even with calibrated probabilities, the confidence of those probabilities alone is not enough to capture the uncertainty of overconfident and erroneous predictions. Figure 17 (a) displays the rejection performance comparing the proposed uncertainty obtained using the wrapper against the predictive entropy using the calibrated probabilities from the original model.
- Compare the wrapper uncertainty using calibrated and non-calibrated probabilities as input for the modelling of the uncertainty. Figure 17 (b) shows the performance

TABLE 2. ECE scores for the different combinations of black-boxes and target applications.

	BB applied to source	BB applied to target	Calibrated BB to target	Wrapper to target
Apply Yelp BB to SST-2	0.0104067	0.150570	0.032845	0.039284
Apply Music BB to Electronics	0.0165734	0.035458	0.035698	0.055868
Apply ImageNet to STL-10	N/A	0.612243	0.206683	0.092530



(a)



(b)

FIGURE 17. Apply SST-2 BB to Yelp with calibrated black-box probabilities for baseline, Apply SST-2 BB to Yelp with calibrated probabilities for training the beta.

of the wrapper trained using the calibrated version of the probabilities.

In both cases, we do not appreciate substantial changes in performance when using and not using calibrated probabilities compared to the uncalibrated version. The only observation that we can remark is the fact that when using the calibrated probabilities as the input for training the wrapper, the training process seems to be smoother than when using the non-calibrated predictions. Moreover, we find that using calibrated probabilities improves the performance of the variation ratios metric in the first stages of the rejection process. Nevertheless, we want to remark that the wrapper itself already has the effect of producing well-calibrated predictions by itself.

IX. CONCLUSIONS AND FUTURE WORK

In this work, we address the problem of accounting for the performance of third-party classification black-boxes when applied to a particular domain. We base the measurement of this reliability on the uncertainty. To measure this uncertainty, we introduced a deep learning wrapper technique that can endow any black-box model with uncertainty features. The wrapper uses a reparameterization trick on the Dirichlet distribution that able to capture the distribution on the multinomial parameters of the output of the black-box classifier.

Differently from previous works that measure uncertainty in Deep Learning models, the novelty of the proposed method relies on the fact that it is able to operate in the constrained setting of pure black-box models.[9] Besides allowing the operation on APIs, or third party close libraries or components, this work is also relevant in use cases where there exist limitations on accessing the resources required for training complex Deep Learning architectures or significant volumes of labelled data.

One advantage of the present work is its ability also to work when the black-box model returns hard predictions, i.e. just the predicted class instead of a distribution over the different classes. Moreover, in addition to the uncertainty estimation for the black-box results, the proposed Dirichlet wrapper incorporates mechanisms to capture the uncertainty even for out-of-distribution points. This feature brings extra-robustness to the proposed method as it will act as a vaccine against future changes in the distribution of the target data set. We also analyse how we can take advantage of the uncertainty modelled to identify issues at the class

[9]Defined by not having access to the internals of the black-box model

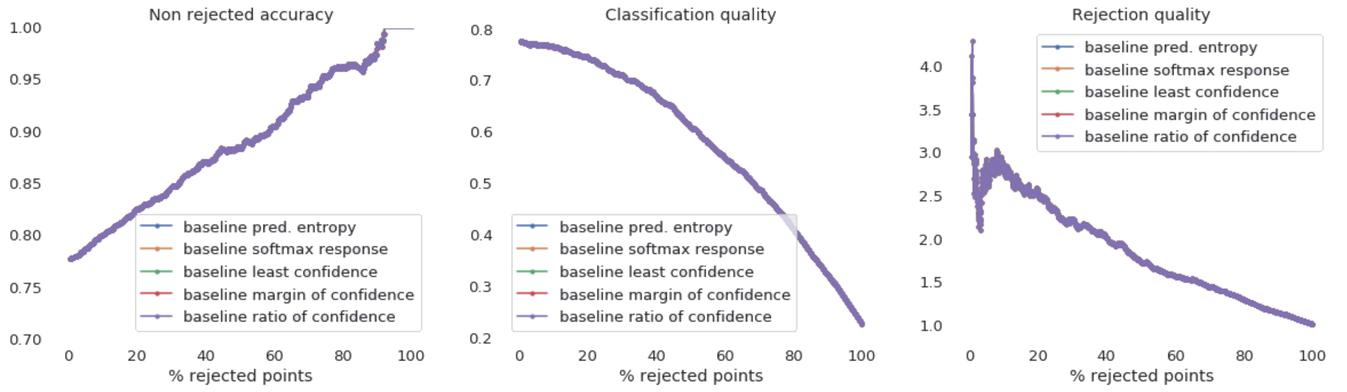


FIGURE 18. Comparative of different baseline metrics that obtain an uncertainty score directly from the black-box prediction, as applied for the Yelp black-box to SST-2 scenario. The baselines compared show similar performance when used for sorting uncertain predictions in the rest of experiments.

level. By analysing the distribution of the uncertainty for each class, we can detect target classes that are problematic for the original model. In this situation, we can decide to omit the unsettled class, try another API or train a model including that class.

To showcase the value of this uncertainty in 34 different problems, including natural language processing and computer vision cases. We use the proposed wrapper to fuel a rejection method and show how this helps in assessing the fitness of a model to a new domain or data set. By measuring the sampling uncertainty and using it for rejection, we can improve the accuracy results in all problems by $4\% - 8\%$ by rejecting only 10% of the samples. Additionally, the method displays a significant value on rejection quality. These results tell us that the predicted uncertainty focuses on intricate, ambiguous, or prone to error cases. The results obtained in both natural language processing and computer vision domains are successful and encouraging.

Besides, we include an in-depth analysis of the method when in front of new classes in the target domain and the relationship with calibration techniques. This analysis highlights the goodness of the proposed method compared with the studied techniques. In particular, regarding calibration, the wrapper already delivers calibrated predictions by itself and shows excellent calibration performance when compared to re-calibration methods.

As future work, we plan to analyse the application of the proposed wrapper beyond the rejection use case presented, such as in the scenario in [49] where they apply uncertainty for acquiring new labels in an active learning process. Additionally, we are evaluating different architectures and strategies for the wrapper implementation and focus on other challenges present in usual cases found in real-life implementations, such as how to deal with high dimensional outputs.

APPENDIX A
COMPARISON OF CONFIDENCE SCORES

In the experiments, we also included other similar metrics derived directly from the original model, namely:

- Predictive entropy, [50], understood as the entropy of the softmax probability distribution.

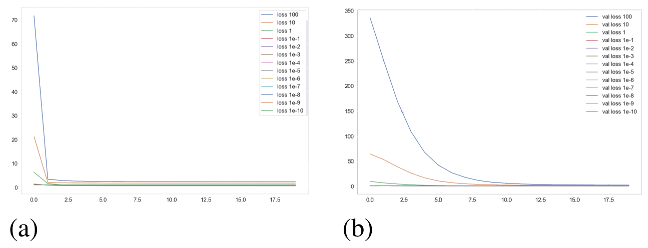


FIGURE 19. Training (a) and validation (b) losses with different λ values.

- Softmax response, as the maximal neuronal response of the softmax layer.
- Least confidence, as defined in [31].
- margin of confidence and ratio of confidence, [32], understood as the margin and ratio respectively between the first and second more confident output units.

As shown in figure 18, in all cases, we observe that all the metrics perform similarly to the entropy, so we chose this uncertainty measure as the baseline to compare with the metrics derived from the wrapper.

APPENDIX B
EFFECT OF THE REGULARIZATION PARAMETER

During the explanation of the Dirichlet Wrapper proposed in this work, we described the loss function used to approximate in equation 9. Recall that in the equation, we introduce a regularisation term controlled with a λ hyper-parameter.

In the present section, we analyse the effect of different values of this λ parameter, studying its impact on the training of the model and the resulting uncertainties. In Figure 19, we display the training and validation losses during the first 20 epochs corresponding to different values of the λ parameter (ranging from 100 to 1e-10). We can observe that for the smaller values of λ, the training is harder and it converges to a higher error minimum. The same happens with higher values than 10.

Figure 20 displays the accuracy obtained when rejecting uncertain scores obtained using different values of λ. We can observe three regimes of performance: for λ values bigger than 1e-5 the resulting rejection systems are rejecting

TABLE 3. The accuracy obtained by training a standalone classifier, applying the API and the proposed wrapper for each domain.

	BB source acc.	BB target acc.	Wrapper target acc.	Non-reject. acc. (10/20/30%)	Class. quality (10/20/30%)	Reject. quality (10/20/30%)
Apply Yelp BB to SST-2	89.18±0.08%	77.13±0.52%	82.62±0.49%	82.43±0.22% 88.19±0.50% 93.60±0.16%	80.04±0.39% 83.11±0.80% 83.05±0.23%	6.03±0.45 6.04±0.51 4.97±0.07
Apply Yelp BB to Amazon apparel		87.86±0.02%	87.68±0.02%	91.37±4e-4% 94.15±2e-3% 95.75±3e-3%	86.18±9e-4% 82.38±3e-3% 75.82±5e-3%	5.25±2.5% 4.18±0.4 3.13±0.2%
Apply Yelp BB to Amazon camera		87.31±0.04%	87.44±0.00%	91.58±7e-4% 94.68±2e-3% 96.62±1e-3%	87.09±1e-3% 83.07±3e-3% 77.57±1e-3%	6.66±4.4 4.84±0.4 3.54±0.0
Apply Yelp BB to Amazon computer		84.32±0.02%	85.58±0.00%	90.24±1e-3% 93.61±3e-3% 96.24±4e-3%	86.16±2e-3% 83.56±6e-3% 78.66±7e-3%	7.29±6.5 5.03±0.7 3.75±0.3
Apply Yelp BB to Amazon electronics		82.52±0.04%	81.87±0.00%	85.39±1e-4% 88.81±1e-3% 92.04±2e-3%	81.06±2e-4% 79.46±3e-3% 76.21±3e-3%	3.54±0.7 3.47±0.2 3.08±0.1
Apply Yelp BB to Amazon health		80.95±0.02%	80.69±0.00%	83.24±6e-4% 86.36±7e-4% 88.70±1e-3%	78.35±1e-3% 76.74±1e-3% 72.78±1e-3%	2.59±0.7 2.81±0.0 2.45±0.0
Apply Yelp BB to Amazon kitchen		83.22±0.04%	82.90±0.00%	86.43±4e-4% 89.47±2e-3% 92.36±2e-3%	82.15±9e-4% 79.75±4e-3% 75.91±4e-3%	4.03±2.8 3.51±0.3 3.02±0.1
Apply Yelp BB to Amazon magazines		83.14±0.03%	86.70±0.01%	88.44±1e-3% 92.33±6e-3% 94.58±6e-3%	85.05±3e-3% 83.59±1e-2% 78.37±1e-2%	7.10±3.2 5.17±1.4 3.62±0.5
Apply Yelp BB to Amazon toys		85.09±0.02%	86.05±0.01%	88.53±5e-3% 91.08±1e-4% 94.08±1e-4%	83.74±1e-3% 81.27±3e-3% 76.15±3e-3%	3.03±0.9 4.43±0.2 3.93±0.1
Apply Yelp BB to Amazon videos		85.91±0.03%	87.65±0.00%	91.75±1e-4% 94.65±2e-3% 96.26±9e-4%	89.15±2e-4% 85.38±4e-3% 78.77±1e-3%	11.92±0.2 5.78±1.0 3.76±0.0
Apply DVD BB to SST-2	89.18±0.08%	78.31±0.03%	81.13±0.00%	82.99±1e-3% 88.03±2e-3% 93.05±5e-3%	80.47±2e-4% 81.92±4e-3% 81.36±8e-3%	5.41±5.6 5.14±0.4 4.41±0.4
Apply DVD BB to apparel	64.87±0.02%	64.73±0.09%	64.38±0.00%	68.19±4e-4% 71.54±1e-3% 74.30±3e-4%	67.83±8e-4% 69.54±2e-3% 69.09±5e-3%	3.37±1.8 2.96±0.1 2.45±0.1
Apply DVD BB to camera		54.25±0.12%	53.62±0.00%	58.21±5e-4% 60.38±1e-3% 62.19±2e-4%	59.94±8e-4% 61.77±2e-3% 62.23±5e-3%	3.59±1.7 2.47±0.3 2.00±0.1
Apply DVD BB to computer		60.54±0.07%	59.96±0.00%	66.04±2e-3% 69.67±5e-3% 71.16±6e-3%	67.56±4e-3% 70.09±9e-3% 68.28±9e-3%	5.76±1.7 3.88±0.3 2.52±0.1
Apply DVD BB to electronics		56.83±0.02%	54.93±0.00%	59.40±1e-4% 62.02±4e-3% 64.69±1e-4%	60.06±2e-4% 62.37±8e-4% 63.70±3e-3%	2.54±1.9 2.31±0.0 2.09±0.0
Apply DVD BB to health		59.82±0.03%	58.69±0.00%	62.61±3e-4% 65.76±3e-3% 68.23±3e-3%	62.69±7e-4% 65.19±5e-4% 65.49±4e-3%	2.59±0.6 2.53±0.3 2.16±0.1
Apply DVD BB to kitchen		61.28±0.03%	60.41±0.00%	63.94±3e-4% 66.52±2e-3% 70.82±3e-3%	63.94±7e-4% 66.52±5e-4% 67.77±4e-3%	2.71±3.8 2.69±0.3 2.45±0.1
Apply DVD BB to magazines		62.79±0.05%	64.16±0.00%	67.38±3e-4% 70.88±2e-3% 74.19±6e-4%	68.04±7e-4% 70.14±4e-4% 70.58±1e-2%	4.65±0.3 3.51±0.4 2.82±0.3
Apply DVD BB to toys		60.76±0.03%	60.19±0.00%	63.83±5e-4% 66.67±2e-3% 69.32±5e-3%	63.94±1e-4% 65.71±3e-4% 66.09±4e-2%	2.87±2.1 2.53±0.2 2.20±0.1
Apply DVD BB to video		64.98±0.03%	66.49±0.00%	69.11±5e-4% 71.09±3e-3% 74.04±5e-3%	69.93±1e-4% 70.05±7e-4% 68.64±8e-2%	4.72±0.1 3.10±0.1 2.37±0.1
Apply Music BB to SST-2	93.08±0.03%	76.65±0.15%	79.71±0.03%	81.61±7e-4% 86.67±1e-3% 92.25±2e-3%	79.67±2e-4% 81.43±5e-3% 81.91±7e-3%	5.85±3.3 5.26±0.6 4.65±0.4
Apply Music BB to apparel	71.88±0.01%	63.68±0.32%	63.31±0.03%	65.53±7e-4% 69.73±1e-3% 72.62±2e-3%	65.90±1e-4% 67.70±1e-3% 67.80±3e-3%	2.69±1.1 2.61±0.1 2.30±0.0
Apply Music BB to camera		59.96±0.09%	60.35±0.00%	63.16±7e-4% 66.66±3e-3% 69.59±2e-3%	63.59±1e-3% 66.54±6e-3% 67.30±3e-3%	3.12±1.3 2.91±0.4 2.45±0.1
Apply Music BB to computer		62.34±0.13%	62.81±0.00%	68.64±2e-3% 72.12±3e-3% 74.92±4e-3%	70.30±4e-3% 72.11±5e-3% 71.60±6e-3%	7.71±1.2 4.28±1.2 3.01±0.3
Apply Music BB to electronics		56.64±0.12%	55.17±0.00%	59.35±0.22% 62.34±0.32% 64.77±0.61%	60.46±0.40% 63.36±0.51% 64.29±0.86%	3.05±0.30 2.67±0.15 2.22±0.13
Apply Music BB to health		58.16±0.26%	57.34±0.00%	61.35±1e-3% 64.09±2e-3% 66.85±3e-3%	62.10±2e-3% 64.19±4e-3% 65.23±5e-3%	3.02±4.2 2.54±0.3 2.23±0.1
Apply Music BB to kitchen		60.53±0.10%	60.29±0.00%	63.76±2e-4% 66.81±3e-3% 70.08±4e-3%	64.16±4e-4% 66.29±5e-3% 67.50±7e-3%	3.25±1.0 2.76±0.4 2.45±0.2
Apply Music BB to magazines		60.96±0.14%	62.06±0.01%	65.07±1e-3% 69.25±4e-3% 72.23±5e-3%	65.77±2e-3% 69.39±7e-3% 69.71±8e-3%	3.95±3.6 3.65±0.6 2.79±0.3
Apply Music BB to toys		60.38±0.11%	60.69±0.00%	63.73±6e-4% 66.44±2e-3% 69.45±4e-3%	64.15±1e-3% 65.73±4e-3% 66.65±6e-3%	3.21±3.5 2.60±0.8 2.32±0.1
Apply Music BB to video		62.73±0.03%	64.27±0.00%	67.17±1e-4% 70.05±2e-3% 72.52±7e-4%	68.13±2e-4% 69.32±4e-3% 68.76±1e-3%	5.62±0.8 3.32±0.3 2.52±0.0
Apply MobileNet V2 ImageNet to STL-10	71.3 [10]	75.73±6e-5%	74.76±0.00%	83.35±1e-3% 87.32±1e-3% 90.55±3e-3%	84.33±1e-3% 84.01±1e-3% 81.07±4e-3%	40.64±0.9 7.54±0.1 4.47±0.1
Apply Inception V3 ImageNet to STL-10	77.9[11]	80.68±1e-5%	80.02±0.00%	88.57±1e-3% 91.11±2e-3% 93.25±1e-3%	88.57±3e-3% 84.92±2e-3% 79.72±2e-3%	33.03±4.8 6.40±0.2 3.91±0.1
Apply DenseNet ImageNet to STL-10	75.0 [12]	79.81±6e-5%	79.69±0.00%	87.85±1e-3% 92.50±8e-4% 95.62±1e-3%	88.15±2e-3% 88.02±1e-3% 83.90±2e-3%	59.62±4.9 9.37±0.1 5.19±0.00

TABLE 4. The accuracy obtained by training a categorical standalone classifier, applying the API and the proposed wrapper for each domain.

	BB source acc.	**BB target acc.**	**Wrapper target acc.**	**Non-reject. acc. (10/20/30%)**	**Class. quality (10/20/30%)**	**Reject. quality (10/20/30%)**
Apply Yelp BB to SST-2	89.18±0.08%	77.13±0.52%	81.14±0.00%	82.38±2e-3% 87.89±3e-3% 93.61±3e-3%	80.27±4e-3% 82.58±6e-3% 83.01±4e-3%	5.87±2.5 5.70±0.4 4.97±0.3
Apply Yelp BB to Amazon apparel		87.86±0.02%	87.60±0.00%	91.01±2e-3% 94.06±2e-3% 96.37±1e-3%	85.54±4e-1% 82.24±3e-3% 76.69±1e-3%	4.62±2.5 4.12±0.1 3.34±0.0
Apply Yelp BB to Amazon camera		87.31±0.04%	86.90±0.00%	90.78±3e-4% 94.79±2e-3% 97.36±8e-3%	85.67±6e-4% 83.94±4e-3% 77.60±1e-3%	5.03±1.9 4.93±0.5 3.81±0.0
Apply Yelp BB to Amazon computer		84.32±0.02%	85.87±0.00%	90.20±1e-3% 92.91±3e-3% 95.92±4e-3%	86.09±2e-3% 82.48±1e-3% 78.23±2e-3%	7.16±4.4 4.55±0.2 3.65±0.1
Apply Yelp BB to Amazon electronics		82.52±0.04%	82.99±0.00%	84.40±1e-4% 88.22±2e-3% 92.86±1e-3%	79.27±2e-4% 78.51±4e-3% 77.37±2e-3%	2.44±0.7 3.15±0.2 3.34±0.1
Apply Yelp BB to Amazon health		80.95±0.02%	80.88±0.00%	83.00±8e-4% 86.34±2e-3% 89.59±2e-3%	77.97±1e-3% 76.72±3e-3% 74.02±3e-3%	2.40±0.7 2.80±0.2 2.68±0.1
Apply Yelp BB to Amazon kitchen		83.22±0.04%	82.90±0.00%	85.84±4e-4% 89.30±9e-3% 92.78±8e-3%	81.11±9e-3% 79.47±1e-2% 76.49±1e-2%	3.41±3.2 3.43±1.4 3.15±0.5
Apply Yelp BB to Amazon magazines		83.14±0.03%	83.35±0.01%	86.03±9e-4% 90.66±2e-3% 95.00±3e-3%	80.79±1e-3% 80.98±4e-3% 78.95±5e-3%	3.24±1.4 4.02±0.2 3.76±0.2
Apply Yelp BB to Amazon toys		85.09±0.02%	85.72±0.00%	88.42±7e-4% 92.20±4e-3% 94.57±1e-3%	83.53±1e-3% 81.91±8e-3% 76.82±2e-3%	4.31±0.8 4.18±0.1 3.27±0.1
Apply Yelp BB to Amazon videos		85.91±0.03%	88.21±0.00%	90.51±1e-4% 96.35±2e-3% 98.29±3e-3%	86.91±2e-4% 88.15±3e-3% 81.61±5e-3%	7.46±4.0 7.64±0.5 4.57±0.4
Apply Music BB to SST-2	89.18±0.08%	78.31±0.03%	80.09±0.00%	81.11±1e-3% 86.34±3e-3% 91.69±3e-3%	78.79±2e-3% 80.90±6e-3% 81.13±6e-3%	4.92±0.6 4.98±0.3 4.40±0.1
Apply Music BB to apparel	64.87±0.02%	64.73±0.09%	63.54±0.00%	66.37±4e-4% 68.91±2e-3% 72.07±2e-3%	65.61±8e-4% 66.39±3e-3% 67.03±4e-3%	2.55±0.8 2.29±0.1 2.18±0.1
Apply Music BB to camera		54.25±0.12%	59.48±0.00%	62.66±4e-3% 66.38±6e-3% 70.45±8e-3%	62.69±7e-3% 66.09±1e-2% 68.51±1e-2%	2.59±0.4 2.80±0.3 2.67±0.2
Apply Music BB to computer		60.54±0.07%	62.37±0.00%	65.88±2e-3% 69.58±6e-3% 72.48±5e-3%	65.47±4e-3% 68.14±1e-2% 62.88±8e-3%	2.83±0.5 2.84±0.2 2.41±0.1
Apply Music BB to electronics		56.83±0.02%	55.83±0.00%	58.76±1e-3% 61.43±3e-3% 64.18±3e-3%	59.38±1e-3% 61.90±5e-3% 63.47±6e-3%	2.39±0.1 2.27±0.0 2.09±0.0
Apply Music BB to health		59.82±0.03%	58.11±0.00%	60.51±1e-3% 63.17±3e-3% 65.80±2e-3%	60.60±3e-4% 62.73±4e-3% 63.78±3e-3%	2.21±0.1 2.18±0.1 2.01±0.1
Apply Music BB to kitchen		61.28±0.03%	59.92±0.00%	63.45±9e-4% 66.62±3e-3% 70.03±1e-3%	63.61±1e-3% 65.99±5e-3% 67.43±2e-3%	2.86±0.0 2.67±0.1 2.54±0.0
Apply Music BB to magazines		62.79±0.05%	60.90±0.00%	64.42±3e-3% 67.80±7e-3% 71.98±9e-3%	64.62±6e-3% 67.10±1e-2% 69.36±1e-2%	3.09±0.3 2.86±0.3 2.73±0.2
Apply Music BB to toys		60.76±0.03%	60.28±0.00%	63.43±4e-4% 66.45±3e-3% 69.31±1e-3%	63.61±6e-3% 65.74±2e-3% 66.45±5e-3%	2.87±0.3 2.59±0.1 2.28±0.0
Apply Music BB to video		64.98±0.03%	63.71±0.00%	67.40±5e-5% 71.25±2e-3% 80.51±4e-3%	68.55±3e-3% 71.23±7e-3% 73.28±8e-2%	6.32±0.7 4.17±0.3 3.51±0.2
Apply DVD BB to SST-2	93.08±0.03%	76.65±0.15%	80.91±0.03%	82.76±2e-3% 87.76±5e-3% 92.59±5e-3%	80.07±5e-3% 81.50±9e-3% 80.71±7e-3%	5.02±0.4 4.94±0.4 4.22±0.1
Apply DVD BB to apparel	71.88±0.01%	63.68±0.32%	64.35±0.03%	67.17±1e-4% 70.16±3e-3% 73.24±4e-3%	66.00±2e-4% 67.34±4e-3% 67.61±5e-3%	2.34±0.1 2.37±0.1 2.21±0.0
Apply DVD BB to camera		54.74±0.09%	54.67±0.00%	57.40±1e-3% 60.16±2e-3% 63.46±4e-3%	58.50±2e-3% 61.43±3e-3% 63.99±4e-3%	2.55±0.1 2.38±0.1 2.26±0.0
Apply DVD BB to computer		62.34±0.13%	60.43±0.00%	64.77±2e-3% 68.00±3e-3% 70.74±4e-3%	65.33±6e-3% 67.49±9e-3% 67.71±1e-2%	3.49±0.4 2.95±0.2 2.43±0.1
Apply DVD BB to electronics		56.64±0.12%	56.14±0.00%	59.24±7e-4% 61.56±2e-3% 64.47±5e-3%	59.77±1e-3% 61.63±4e-3% 63.38±7e-3%	2.39±0.1 2.14±0.1 2.05±0.1
Apply DVD BB to health		58.16±0.26%	58.94±0.00%	61.98±2e-3% 64.48±3e-3% 67.14±6e-3%	61.56±3e-3% 63.16±5e-3% 63.98±9e-3%	2.07±0.1 2.06±0.1 1.96±0.1
Apply DVD BB to kitchen		60.53±0.10%	60.79±0.00%	64.14±1e-3% 67.31±3e-3% 70.51±4e-3%	64.10±3e-3% 66.33±5e-3% 67.35±6e-3%	2.78±0.1 2.64±0.1 2.38±0.0
Apply DVD BB to magazines		60.96±0.14%	62.60±0.01%	66.97±3e-3% 70.90±3e-3% 74.23±1e-3%	67.31±1e-3% 70.17±6e-3% 70.65±4e-3%	3.99±0.4 3.50±0.5 2.83±0.1
Apply DVD BB to toys		60.38±0.11%	60.15±0.00%	63.67±3e-3% 66.30±3e-3% 68.61±4e-3%	63.67±5e-3% 65.12±4e-3% 65.10±6e-3%	2.73±0.3 2.38±0.1 2.06±0.1
Apply DVD BB to video		62.73±0.03%	64.98±0.00%	69.71±5e-4% 73.40±2e-3% 77.75±2e-3%	70.45±1e-3% 72.41±3e-3% 73.82±3e-3%	6.23±0.1 4.03±0.1 3.40±0.1
Apply MobileNet V2 ImageNet to STL-10	71.3 [13]	75.56±6e-5%	74.62±0.00%	82.22±9e-4% 86.19±9e-4% 88.78±3e-3%	82.29±1e-3% 82.20±1e-3% 78.60±1e-4%	15.07±0.1 6.12±0.1 3.78±0.1
Apply Inception V3 ImageNet to STL-10	77.9[14]	80.68±6e-5%	80.55±0.00%	86.99±2e-3% 89.57±1e-3% 91.13±2e-3%	85.73±4e-3% 82.47±2e-3% 76.76±3e-3%	12.46±1.6 4.98±0.1 3.21±0.1
Apply DenseNet ImageNet to STL-10	75.0 [15]	79.81±6e-5%	78.87±0.00%	86.88±1e-3% 89.29±2e-3% 90.79±1e-3%	86.41±4e-3% 82.91±2e-3% 77.15±3e-3%	18.43±1.3 5.38±0.1 3.31±0.0

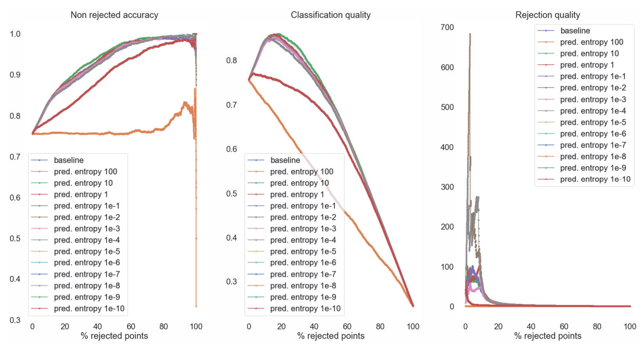
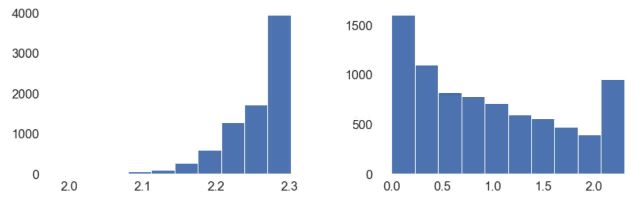


FIGURE 20. Rejection with the uncertainty scores computed using different λ values.



(a) $\lambda = 1e - 2$ (b) $\lambda = 1e - 2$

FIGURE 21. Uncertainty scores distribution with different λ values.

uncertain and erroneous points better than the baseline. On the other hand, for smaller values, below 1e-5, we observe no difference with the selected baseline. Finally, high values of λ, e.x. 100, perform even worse than the baseline, showing a total degradation of the uncertainty model. We also can observe that the higher the λ value, the better it sorts the erroneous predictions.

Even though high values of λ seem to perform better than tiny ones, we recommend using values between 1e-2 and 1e-3. Figure 21 show the distribution of the uncertainty scores when $\lambda = 10$ and $\lambda = 1e - 2$ respectively. In the case of higher values of the λ, the model seems to consider everything to be uncertain. The explanation is that by assigning high values for the regularizer, the value of the β parameter tends to be a tiny number. When multiplying the β by the original prediction, the resulting Dirichlet control parameters, α, are infinitesimal, and they cause a distribution with very high entropy. On the contrary, in the case of 1e-2, we obtain similar performance in the rejection but with more interpretable uncertainty scores.

Even though the lambda parameter supposes a new hyper-parameter to tune, the range of values used in all the experiments did not vary from $1e - 1$ to $1e - 2$. As an initial guess in potential new applications of the method, a sensible choice could be starting by using a value of $1e - 2$.

APPENDIX C
SUMMARY TABLE FOR NLP PROBLEMS USING PROBABILISTIC OUTPUTS

Table 3 displays the detail of the results obtained in all the experiments run using the probabilistic output of the black-boxes.

APPENDIX D
SUMMARY TABLE FOR NLP PROBLEMS USING CATEGORICAL OUTPUTS

Table 4 displays the detail of the results obtained in all the experiments run using the categorical output of the black-boxes.

REFERENCES

[1] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad, "Intelligible models for HealthCare: Predicting pneumonia risk and hospital 30-day readmission," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA, 2015, pp. 1721–1730, doi: 10.1145/2783258.2788613.

[2] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," *CoRR*, vol. abs/1604.07316, 2016. [Online]. Available: http://arxiv.org/abs/1604.07316

[3] J. M. Hernandez-Lobato and R. Adams, "Probabilistic backpropagation for scalable learning of Bayesian neural networks," in *Proc. 32nd Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 37, F. Bach and D. Blei, Eds. Lille, France: PMLR, Jul. 2015, pp. 1861–1869. [Online]. Available: http://proceedings.mlr.press/v37/hernandez-lobatoc15.html

[4] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, Inc., 2016, pp. 1019–1027.

[5] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5574–5584.

[6] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, Dept. Eng., Univ. Cambridge, Cambridge, U.K., 2016.

[7] L. A. F. Fernandes and M. M. Oliveira, "Handling uncertain data in subspace detection," *Pattern Recognit.*, vol. 47, no. 10, pp. 3225–3241, Oct. 2014.

[8] J. Gast and S. Roth, "Lightweight probabilistic deep networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3369–3378.

[9] P. Sadowski and P. Baldi. (2019). *Neural Network Regression with Beta, Dirichlet, and Dirichlet-Multinomial Outputs*. [Online]. Available: https://openreview.net/forum?id=BJeRg205Fm

[10] A. Malinin and M. Gales, "Predictive uncertainty estimation via prior networks," in *Proc. Adv. Neural Inf. Process. Syst.*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2018, pp. 7047–7058. [Online]. Available: http://papers.nips.cc/paper/7936-predictive-uncertainty-estimation-via-prior-networks.pdf

[11] W. Chen, Y. Shen, X. Wang, and W. Y. Wang, "Enhancing the robustness of prior network in out-of-distribution detection," *CoRR*, vol. abs/1811.07308, 2018. [Online]. Available: http://arxiv.org/abs/1811.07308

[12] C. Chow, "On optimum recognition error and reject tradeoff," *IEEE Trans. Inf. Theory*, vol. 16, no. 1, pp. 41–46, Jan. 1970.

[13] L. P. Cordella, C. De Stefano, F. Tortorella, and M. Vento, "A method for improving classification reliability of multilayer perceptrons," *IEEE Trans. Neural Netw.*, vol. 6, no. 5, pp. 1140–1147, Sep. 1995.

[14] C. De Stefano, C. Sansone, and M. Vento, "To reject or not to reject: That is the question-an answer in case of neural classifiers," *IEEE Trans. Syst., Man Cybern., C (Appl. Rev.)*, vol. 30, no. 1, pp. 84–94, Feb. 2000.

[15] R. El-Yaniv and Y. Wiener, "On the foundations of noise-free selective classification," *J. Mach. Learn. Res.*, vol. 11, pp. 1605–1641, May 2010.

[16] Y. Geifman and R. El-Yaniv, "Selective classification for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4878–4887.

[17] P. L. Bartlett and M. H. Wegkamp, "Classification with a reject option using a hinge loss," *J. Mach. Learn. Res.*, vol. 9, pp. 1823–1840, Jun. 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1390681.1442792

[18] C. Cortes, G. DeSalvo, and M. Mohri, "Boosting with abstention," in *Proc. Adv. Neural Inf. Process. Syst.*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, 2016, pp. 1660–1668. [Online]. Available: http://papers.nips.cc/paper/6336-boosting-with-abstention.pdf

[19] S. Thulasidasan, T. Bhattacharya, J. Bilmes, G. Chennupati, and J. Mohd-Yusof, "Combating label noise in deep learning using abstention," in *Proc. 36th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 97, K. Chaudhuri and R. Salakhutdinov, Eds. Long Beach, CA, USA: PMLR, Jun. 2019, pp. 6234–6243. [Online]. Available: http://proceedings.mlr.press/v97/thulasidasan19a/thulasidasan19a.pdf

[20] S. Shekhar, M. Ghavamzadeh, and T. Javidi, "Binary classification with bounded abstention rate," 2019, *arXiv:1905.09561*. [Online]. Available: http://arxiv.org/abs/1905.09561

[21] Y. Geifman and R. El-Yaniv, "SelectiveNet: A deep neural network with an integrated reject option," in *Proc. 36th Int. Conf. Mach. Learn. (ICML)* (Proceedings of Machine Learning Research), vol. 97, K. Chaudhuri and R. Salakhutdinov, Eds. Long Beach, CA, USA: PMLR, Jun. 2019, pp. 2151–2159. [Online]. Available: http://proceedings.mlr.press/v97/geifman19a.html

[22] M. S. A. Nadeem, J.-D. Zucker, and B. Hanczar, "Accuracy-rejection curves (ARCs) for comparing classification methods with a reject option," in *Proc. Mach. Learn. Syst. Biol.*, 2009, pp. 65–81.

[23] T. C. W. Landgrebe, D. M. J. Tax, P. Paclík, and R. P. W. Duin, "The interaction between classification and reject performance for distance-based reject-option classifiers," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 908–917, Jun. 2006.

[24] I. Pillai, G. Fumera, and F. Roli, "Multi-label classification with a reject option," *Pattern Recognit.*, vol. 46, no. 8, pp. 2256–2266, Aug. 2013, doi: 10.1016/j.patcog.2013.01.035.

[25] B. Hanczar, "Performance visualization spaces for classification with rejection option," *Pattern Recognit.*, vol. 96, Dec. 2019, Art. no. 106984.

[26] F. Condessa, J. Bioucas-Dias, and J. Kovačević, "Performance measures for classification systems with rejection," *Pattern Recognit.*, vol. 63, pp. 437–450, Mar. 2017.

[27] Y. Xiao and W. Y. Wang, "Quantifying uncertainties in natural language processing tasks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 7322–7329.

[28] J. Mena, O. Pujol, and J. Vitrià, "Dirichlet uncertainty wrappers for actionable algorithm accuracy accountability and auditability," 2019, *arXiv:1912.12628*. [Online]. Available: http://arxiv.org/abs/1912.12628

[29] K. Lee, H. Lee, K. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–4. [Online]. Available: https://openreview.net/forum?id=ryiAv2xAZ

[30] L. Freeman, *Elementary Applied Statistics: For Students in Behavioral Science*. Hoboken, NJ, USA: Wiley, 1965. [Online]. Available: https://books.google.es/books?id=r4VRAAAAMAAJ

[31] A. Culotta and A. McCallum, "Reducing labeling effort for structured prediction tasks," in *Proc. 20th Nat. Conf. Artif. Intell.*, vol. 2, 2005, pp. 746–751. [Online]. Available: http://dl.acm.org/citation.cfm?id=1619410.1619452

[32] T. Scheffer, C. Decomain, and S. Wrobel, "Active hidden Markov models for information extraction," in *Proc. 4th Int. Symp. Intell. Data Anal.*, London, U.K.: Springer-Verlag, 2001, pp. 309–318. [Online]. Available: http://dl.acm.org/citation.cfm?id=647967.741626

[33] L. Mou, Z. Meng, R. Yan, G. Li, Y. Xu, L. Zhang, and Z. Jin, "How transferable are neural networks in NLP applications?" in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1–10, doi: 10.18653/v1/d16-1046.

[34] S. Tan, X. Cheng, Y. Wang, and H. Xu, "Adapting naive Bayes to domain adaptation for sentiment analysis," in *Advances in Information Retrieval*, M. Boughanem, C. Berrut, J. Mothe, and C. Soule-Dupuy, Eds. Berlin, Germany: Springer, 2009, pp. 337–349.

[35] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proc. 28th Int. Conf. Int. Conf. Mach. Learn.*, New York, NY, USA: Omnipress, 2011, pp. 513–520. [Online]. Available: http://dl.acm.org/citation.cfm?id=3104482.3104547

[36] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification," in *Proc. 45th Annu. Meeting Assoc. Comput. Linguistics*. Prague, Czech Republic, Jun. 2007, pp. 440–447. [Online]. Available: https://www.aclweb.org/anthology/P07-1056

[37] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Seattle, WA, USA, Oct. 2013, pp. 1631–1642. [Online]. Available: https://www.aclweb.org/anthology/D13-1170

[38] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[39] Y. Chen, C. Yang, Y. Zhang, and Y. Li, "Deep conditional adaptation networks and label correlation transfer for unsupervised domain adaptation," *Pattern Recognit.*, vol. 98, Feb. 2020, Art. no. 107072. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320319303735

[40] J. Liang, R. He, Z. Sun, and T. Tan, "Exploring uncertainty in pseudo-label guided unsupervised domain adaptation," *Pattern Recognit.*, vol. 96, Dec. 2019, Art. no. 106996. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320319302997

[41] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[42] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.

[43] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[44] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[45] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. 14th Int. Conf. Artif. Intell. Statist.* (Proceedings of Machine Learning Research), vol. 15, G. Gordon, D. Dunson, and M. Dudík, Eds. Fort Lauderdale, FL, USA: PMLR, Apr. 2011, pp. 215–223. [Online]. Available: http://proceedings.mlr.press/v15/coates11a.html

[46] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *Proc. 32nd Annu. Meeting Assoc. Comput. Linguistics*, 1994, pp. 133–138.

[47] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. 34th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 70, D. Precup and Y. W. Teh, Eds. Sydney, NSW, Australia: PMLR, Aug. 2017, pp. 1321–1330. [Online]. Available: http://proceedings.mlr.press/v70/guo17a.html

[48] M. P. Naeini, G. Cooper, and M. Hauskrecht, "Obtaining well calibrated probabilities using Bayesian binning," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2901–2907.

[49] J. Liang, R. He, Z. Sun, and T. Tan, "Exploring uncertainty in pseudo-label guided unsupervised domain adaptation," *Pattern Recognit.*, vol. 96, Dec. 2019, Art. no. 106996.

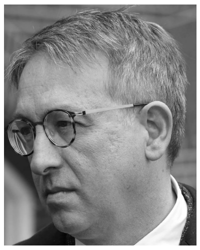
[50] I. Dagan and S. P. Engelson, "Committee-based sampling for training probabilistic classifiers," in *Proc. 12th Int. Conf. Int. Conf. Mach. Learn.* San Francisco, CA, USA: Morgan Kaufmann, 1995, pp. 150–157. [Online]. Available: http://dl.acm.org/citation.cfm?id=3091622.3091641

JOSÉ MENA was born in Barcelona, Spain, in 1975. He received the Technical Engineer degree in computer systems from Universitat Rovira i Virgili and the Computer Science degree from the Universitat Oberta de Catalunya (UOC). He is currently pursuing the Ph.D. degree in apply deep learning to recommender systems with the Universitat de Barcelona and the Doctoral Program in mathematics and computing.

Since 2012, he has been a Researcher with the Big Data and Data Science Unit, Eurecat, Catalan Technology Centre. Previously to Eurecat, he worked for more than ten years on consultancy companies like T-systems or Opentrends developing projects for public administrations and private companies, always working with open source technologies. He has experience with big data technologies, from NoSQL technologies to distributed computing frameworks like spark, machine learning algorithms and libraries and cloud platforms like Amazon web services. In the last five years, during his work at Eurecat, he has gained experience in R&D European and National projects, focused on recommender systems for mobility and tourism domains, complemented with innovation projects for private companies.

He was a recipient of a Scholarship from AGAUR of the Generalitat de Catalunya through the Industrial Ph.D. grant program.

ORIOL PUJOL received the Ph.D. degree from the Universitat Autònoma de Barcelona, in 2004, on a work on deformable models applied to medical imaging and fusion of supervised and unsupervised learning. He is currently Associate Professor with the Department of Mathematics and Computer Science at Universitat de Barcelona. In recent years, his research has focused on the mathematical and algorithmic side of supervised machine learning techniques, including ensemble learning methods, kernel machines, online methods, deep learning, and probabilistic methods. He also has experience in the applications of machine learning to problems in finance, ehealth, medical imaging, wearable sensors, autonomous driving systems, marketing, and he is lately focused on algorithmic auditing techniques.



JORDI VITRIÀ is currently a Full Professor from the Universitat de Barcelona (UB), which he joined in 2007, and where he teaches an introductory course on Algorithms and advanced courses on Data Science and Deep Learning. From April 2011 to January 2016, he served as the Head of the UB's Applied Mathematics and the Analysis Department. He is also a member of the Mathematics and Computer Science Department, UB. His research, when personal computers had 128KB of memory, was originally oriented towards digital image analysis and how to extract quantitative information from them, but soon evolved towards computer vision problems. After a Postdoctoral year at the University of California at Berkeley, in 1993, he focused on Bayesian methods for computer vision methods. He is leading a research group working in deep learning, computer vision, and machine learning. He has authored more than 100 peer-reviewed articles and holds eight international patents. He has directed 14 Ph.D. theses in the area of machine learning and computer vision. He has been the leader of a large number of research projects at international and national level.