A High-Probability Safety Guarantee for Shifted Neural Network Surrogates

Mélanie Ducoffe[1*], **Sébastien Gerchinovitz**[1,2], **Jayant Sen Gupta**[1*]

[1] IRT Saint Exupéry, Toulouse, France
{melanie.ducoffe,sebastien.gerchinovitz, jayant.sen-gupta}@irt-saintexupery.com
[2]Institut de Mathématiques de Toulouse, UMR5219, Université de Toulouse, CNRS, France

Abstract

Embedding simulation models developed during the design of a platform opens a lot of potential new functionalities but requires additional certification. Usually, these models require too much computing power, take too much time to run so we need to build an approximation of these models that can be compatible with operational constraints, hardware constraints, and real-time constraints. Also, we need to prove that the decisions made by the system using the surrogate model instead of the reference one will be safe. The confidence in its safety has to be demonstrated to certification authorities. In cases where safety can be ensured by systematically over-estimating the reference model, we propose different probabilistic safety bounds that we apply on a braking distance use-case. We also derive a new loss function suited for shifted surrogates and study the influence of the different confidence parameters on the trade-off between the safety and accuracy of the surrogate models. Here are the main contributions and the outline of this paper:

- We define safety as the fact that a surrogate model should over-estimate the reference model with high probability.
- We use Bernstein-type deviation inequalities to estimate the probability of under-estimating a reference model with a surrogate model.
- We show how to shift a surrogate to guarantee safeness with high probability.
- Since shifting impacts the performance of our surrogate, we derive a new regression loss function—that we call SMSE—in order to build surrogates with safeness-promoting constraints.

Introduction

Deep Learning has undoubtedly provided tremendous progress in many machine learning tasks. More specifically, it has improved the state of the art in regression problems (Lathuilière et al. 2019) and demonstrated, in our industrial applications, that it achieved better accuracy. However, one limitation that restrains their massive industrialization nowadays for safety-critical tasks is mainly due to the limited confidence in their prediction. Indeed, neural networks usually output only point prediction without any calibrated estimate of their uncertainty. This is highly problematic in high stakes scenarios. When it comes to the aircraft industry, aeronautical certification authorities have already identified the need for AI to refrain from responding rather than transmitting erroneous information (FAA 2016). Consequently, uncertainty assessment and prediction confidence intervals will become necessary in a certification process of AI for critical systems. Indeed, the error of design that always comes within any machine learning model is a novelty into the certification process and thus not taken into account within the DO-178c (Hilderman 2014). Eventually, it implies that these uncertainties will have to be calibrated and quantified. In practice, the uncertainty may result from several sources, as also stated in the taxonomy of (De Rocquigny et al. 2008) and (Der Kiureghian and Ditlevsen 2009).

In this work, we are interested in quantifying the uncertainty of a neural network surrogate as a proxy for a physical simulator. Indeed, in many industries, including aeronautics, numerical simulators have been developed to model physical phenomena inherent in their systems (Biannic et al. 2016). As these models are based on physical equations, whose relevancy is asserted by scientific experts, their qualification will have or can be carried out without any issue. Since their computational costs and running time prevent us from embedding them on board, the use of these simulators in the aeronautical field remains mainly limited to the development and design phase of the aircraft. Thanks to the current success of deep neural networks, previous works have already investigated neural network based surrogates for approximating numerical simulators (Jian et al. 2017; Sudakov et al. 2019).

We are convinced that these surrogates position us in a very favorable framework towards a certification process. First of all, we should not demonstrate the ability of our surrogate to model a real phenomenon, but its quality to approximate the simulator itself: the simulator is the reference. Nevertheless, the simulator outputs a deterministic target for any input data on a bounded domain, without any notion of how likely the input configuration is in practice. Concerning the training data, we consider a uniform distribution as we want our surrogate model to be well trained on the whole support of inputs and not wish to disregard cases that could

[*] seconded from Airbus AI Research, Toulouse

be rare in real life. It should be noted, however, that the sampled space must encompass the real space of observations without it being oversized.

These early conclusions do not in any way mean that we currently have the necessary tools to certify such surrogates. Other difficulties have to be taken into account. First of all, an average error will not be sufficient for a certification context, because it reflects a global behavior whereas current certification standards are based on the notion of the worst-case scenario. An example is the certification of hardware by upper-bounding the worst-case running time, as described in (Wilhelm et al. 2008).

Recent works have proposed to construct prediction intervals with high probability confidence, conditioned on the input data (Pearce et al. 2018) and (Tagasovska and Lopez-Paz 2019). However, the significance level accepted in critical systems is far below the ones reported in the literature for this kind of method. Definitively, these methods remain interesting in a trustworthy approach but will not be relevant to provide guarantees tight enough for aeronautical certification. Their limitation mainly lies in the fact that they fit a loss function during the training of the neural network, which is trained on mini-batches whose size affects the level of significance on which we can train our model. These methods will be further described in the Related Works section. Although it can easily be assumed that it is possible to collect an unlimited number of simulated data, the economic cost will be a limiting factor in the long-term establishment of certification for data-based surrogates.

Another property rather specific to these surrogates is that under or over-estimating the target will not have the same impact on the system's safety. Consider a surrogate of the landing distance of an aircraft. Under-estimating the reference distance could lead to an avoidable overrun. On the contrary, over-estimating the reference distance may lead to unnecessary turn-around maneuvers when the landing was indeed safely possible. These different scenarios are summarized and illustrated in Figure 1. Eventually, we are more interested in asymmetric guarantees: if the second failure scenario will waste money for the airline, the first failure scenario is a threat to the safety and should be avoided in any case for the sake of the certification of the system. Based on this reasoning, we define the notion of **safe surrogate** as a surrogate that over-approximates its reference model with high probability.

In the next sections, we will show how we can establish better confidence probabilities for safe surrogates than the significance level optioned in the literature.

Related Works

Most machine learning methods that tackle regression tasks use a symmetric loss function to train their surrogates. However, there exist asymmetric loss functions to penalize either under-estimation or over-estimation given the context of use (Yao and Tong 1996). The use of such functions have been compared in the case of predictive maintenance, but to the best of our knowledge, no theoretical guarantees for the safety of the surrogate have been provided (Tolstikov, Janssen, and Fürnkranz 2016).

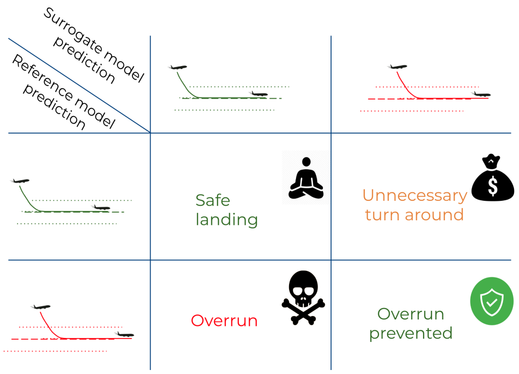


Figure 1: Impact of over-estimating the landing distance of an aircraft. Cases are "runway is long enough" and "runway is too short", rows correspond to reference model output, columns correspond to surrogate model output

A whole part of the literature relies on probabilistic assumptions and provide guarantees at the condition that such assumptions are verified. In this tendency, we can cite Bayesian modeling. Bayesian Neural Networks produce a probabilistic relationship between the network input and output but based on assumptions on the structure dependence of the random vector of weights of the network (Gal and Ghahramani 2015). The problem with this type of approach is that it intrinsically depends on the initial assumption which is difficult to validate or belie.

Since the performance of deep neural networks are widely known for classification tasks, (Keren, Cummins, and Schuller 2018) cast a regression problem with uncertainty as a classification task. To do so, they split the ranges of output values into chunks, and attribute labels to each chunk. When the prediction of a sample falls into a chunk, first it provides an upper and a lower bound of the ground-truth target, but also a notion of confidence on this prediction. They use *distillation* to balance the predicted confidence and the empirical uncertainty (Papernot et al. 2016). Note that cross-entropy is not the best loss function in this context as it penalizes equally misclassification, without any notion of distance, and sign of the errors; which is not adequate for training a safe surrogate. Probably, using Wasserstein loss would be more suitable (Frogner et al. 2015)

Next, we describe state-of-the-art methods to incorporate uncertainties into the training of neural networks for regression problem.

High Quality Prediction Interval for Deep Learning In (Pearce et al. 2018), Pearce et al. called their method QD, a neural network that outputs a lower and an upper bound of the prediction interval. In order to train the network, it first expresses the prediction interval as the combination of two uncertainty-based factors: PICP and MPIW:

- *Prediction Internal Coverage Probability*: given a test dataset, PICP is the average number of points that fall between the predicted lower and upper bounds.

- *Mean Prediction Interval Width*: MPIW is the average distance between the predicted upper and lower bounds.

Based on this reasoning, they derive a training loss as a weighted linear combination of these two costs.

Intuitively, QD must minimize the width of the prediction interval, MPIW, under the assumption that most of the predicted intervals are correct. Indeed, QD has a hyperparameter α such that $(1-\alpha)$ represents the level of significance. Unlike in the previous work of (Khosravi et al. 2010), QD minimizes MPIW subject to the samples that fall into their predicted interval, so not to shrink further points badly predicted.

$$Loss_{QD} = MPIW + \lambda \frac{n}{\alpha(1-\alpha)} \max(0, (1-\alpha) - PICP)^2 \quad (1)$$

Although QD is motivated by theoretical intuitions, several underlying assumptions can be discussed in a certification context:

- The training may not converge to the given significance level for very low α.
- The predicted interval for the training samples are biased by the optimization procedure, thus the performance should be validated on a test set.
- They assume that the probability of two samples to be covered by their predicted interval is independent, which may not be the case in practice when data points are similar. This assumptions is essential in their formulation as it allows them to approximate the total number of points to be rightly covered by a binomial distribution.
- Since the authors encounter some unstability issue by optimizing directly PICP, they propose a soft approximation to be trained on.

The previous flaws can be argued and questioned, and they can probably be put into perspective. Nevertheless, we argue that QD cannot scale to really high significance level, unless scaling the size of the minibatch accordingly. As explained previously, gathering many training samples will be a limiting factor in the long-term establishment of certification for data-based surrogates.

Single Model Uncertainties for Deep Learning The goal of quantile regression is to estimate the conditional quantiles $F^{-1}(\tau \mid X = x)$ of a real-valued random variable Y given another random variable X, for some quantile level τ. In (Tagasovska and Lopez-Paz 2019) the authors propose the SQR method to estimate the quantile distribution function $x \mapsto F^{-1}(\tau \mid X = x)$ with a neural network surrogate $\widehat{y} = \widehat{f}_\tau(x)$. To that aim, they train their surrogate to minimize the pinball loss:

$$\ell_\tau(y, \widehat{y}) = \left\{ \begin{array}{cc} \tau(y - \widehat{y}) & \text{if } y - \widehat{y} \geq 0 \\ (1-\tau)(\widehat{y} - y) & \text{otherwise} \end{array} \right. \quad (2)$$

It is well known that the pinball loss is such that $\widehat{y} \mapsto \mathbb{E}\big[\ell_\tau(Y, \widehat{y}) \,|X\big]$ is minimized when $\widehat{y}$ is equal to the level-τ quantile of the conditional distribution of Y given X.

The authors train a surrogate with the pinball loss with a random threshold τ to perform quantile regression simultaneously for all quantile levels. No theoretical analysis however quantifies how the outputs of the trained neural network are close to the conditional quantiles of Y given X.

A Provably Safe Shifted Surrogate

The core value of this contribution is to enforce that a surrogate is safe with high confidence, rather than how to obtain such a surrogate. The design of a safe surrogate will be analysed further in the next sections.

Without any prior knowledge on the reference model (such as its smoothness), it is virtually impossible to prove that a given surrogate is safe in a worst-case sense. It is however possible to control the probability of under-estimating the output of the reference model without any assumptions on it. We first define the notion of *safeness* below.

Definition 1 (Safe Surrogate) *Let $\varepsilon \in (0,1)$. Given a reference model $f : \mathcal{X} \to \mathbb{R}$ and a probability distribution P_X on the domain $\mathcal{X}$, we say that a surrogate $\widehat{f} : \mathcal{X} \to \mathbb{R}$ is $(1-\varepsilon)$-**safe** if it over-approximates the reference model f with probability at least $1-\varepsilon$, i.e.,*

$$\mathbb{P}\left(\widehat{f}(X) \geq f(X)\right) \geq 1 - \varepsilon$$

or, equivalently,

$$\mathbb{P}\left(f(X) > \widehat{f}(X)\right) \leq \varepsilon \ .$$

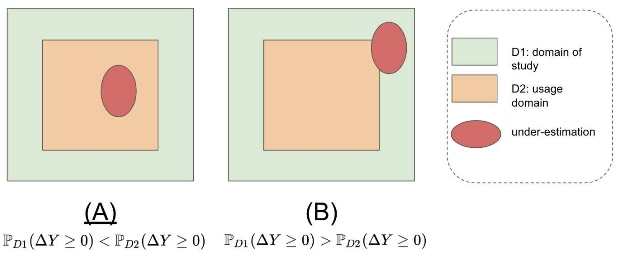
The two probabilities above are taken with respect to a random variable X drawn from the distribution P_X, but $\widehat{f}$ is considered as fixed. [1]

In practice, if the distribution P_X is chosen to be uniform on a bounded subset $\mathcal{S} \subset \mathcal{X}$ (called the *domain of study* thereafter), then the probability $\mathbb{P}\big(f(X) > \widehat{f}(X)\big)$ is the proportion of all possible configurations x for which the surrogate $\widehat{f}$ underestimates the reference model's values $f(x)$. Such problematic event is represented in red on Figure 2.

We now point to two limitations of such high-probability safety guarantees. First, since both our surrogate and our reference models are deterministic, underestimation will always happen in the under-estimation region $\{f > \widehat{f}\}$. Second, as shown by the toy illustrative cases A and B above, the choice of the domain of study may impact the value of the under-estimation probability $\mathbb{P}\big(f(X) > \widehat{f}(X)\big)$. Indeed the domain of study should contain the usage domain (all typical configurations), but taking it much larger may decrease (case A) or increase (case B) the weight of the under-estimation region. As a consequence, high-probability guarantees should be interpreted carefully and used in conjunction with a proper choice of the distribution P_X.

Certification authorities will most likely require safeguards. Without being exhaustive, among the justifications that will be necessary, we mention the following:

[1] More formally, if $\widehat{f}$ is constructed using a training set, then these probabilities are conditional probabilities given the training set. We look for guarantees that hold for every realization of $\widehat{f}$.



(A) $\mathbb{P}_{D1}(\Delta Y \geq 0) < \mathbb{P}_{D2}(\Delta Y \geq 0)$

(B) $\mathbb{P}_{D1}(\Delta Y \geq 0) > \mathbb{P}_{D2}(\Delta Y \geq 0)$

Figure 2: A toy illustration of the meaning of high-probability safety guarantees, where $\Delta Y = f(X) - \widehat{f}(X)$. Taking a larger domain of study than the usage domain may decrease (case A) or increase (case B) the weight of the under-estimation region.

- The domain of study must contain the usage domain, without it being oversized, in order to avoid underestimating the risk of underestimation in the real usage domain.
- Ultimately, the surrogate should be validated on real scenarios (such as flight tests).

Next we explain how to compute a certified upper bound on the probability $\mathbb{P}\big(f(X) > \widehat{f}(X)\big)$. We follow a very natural Monte-Carlo approach: we sample random points $X_1, \ldots, X_n$ independently from the same distribution P_X in the domain of study, and count how many errors $f(X_i) - \widehat{f}(X_i)$ are positive. Importantly, the test dataset $X_1, \ldots, X_n$ is independent from the training set that was used to build the surrogate $\widehat{f}$.

Bernstein's inequality to assess safeness

To assess how safe is a given surrogate $\widehat{f}$ given n observations $f(X_i)$ and $\widehat{f}(X_i)$, we use a probabilistic deviation inequality known as Bernstein's inequality (Bernstein 1924). The version we state below is a direct consequence of Theorem 2.10 in (Boucheron, Lugosi, and Massart 2013), instantiated with the random variables $-\mathbb{1}_{f(X_i) > \widehat{f}(X_i)}$, $1 \leq i \leq n$, and the parameters $v = n\mathbb{P}(f(X) > \widehat{f}(X))$ and $c = 1/3$. The constant 3.15 could probably be slightly improved.

Proposition 1 (Consequence of Bernstein's inequality) *Consider $n \geq 2$ independent random variables $X_1, \ldots, X_n$ drawn from the same distribution P_X in the domain of study, and independent of the training set ($\widehat{f}$ is considered as fixed). We estimate the under-estimation probability by*

$$\widehat{G}_n = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{f(X_i) > \widehat{f}(X_i)}$$

Then, for any risk level $\delta \in (0, 1)$, the following inequality holds with probability at least $1 - \delta$ over the choice of the test set $X_1, \ldots, X_n$:

$$\mathbb{P}\big(f(X) > \widehat{f}(X)\big) \leq \widehat{G}_n + \sqrt{\frac{2\widehat{G}_n}{n} \ln\left(\frac{1}{\delta}\right)} + \frac{3.15}{n} \ln\left(\frac{1}{\delta}\right)$$

The above result means that, over all possible choices of the test set $X_1, \ldots, X_n$, a large proportion $1 - \delta$ of them allows us to upper bound the probability $\mathbb{P}\big(f(X) > \widehat{f}(X)\big)$ by the observed probability $\widehat{G}_n$ plus a small remainder term proportional to $\sqrt{\widehat{G}_n \ln(1/\delta)/n} + \ln(1/\delta)/n$.

Note also that when we observe $\widehat{G}_n = 0$, we have a high-confidence proof that our surrogate is $(1 - \varepsilon)$-safe, with $\varepsilon = 3.15 \ln(1/\delta)/n$. However, we get a non-negligible value for ε whenever $\widehat{G}_n$ is bounded away from zero. A simple way to make $\widehat{G}_n$ smaller is to shift the surrogate predictions $\widehat{f}(X)$ by a non-negative amount t. In the next paragraphs we study how to choose t in order to get a provably small upper-bound on $\mathbb{P}\big(f(X) > \widehat{f}(X) + t\big)$.

Shifted Surrogate

As mentioned above, we propose to consistently shift the predictions of the surrogate $\widehat{f}$ by a non-negative real number t, in order to make it $(1 - \varepsilon)$-safe with a small value of ε. More precisely, we consider the notion of *shifted surrogate* defined below, and illustrated on Figure 3. The main theoretical guarantee will be stated in Corollary 1 below.

Definition 2 (Shifted Surrogate) *Given a surrogate $\widehat{f} : \mathcal{X} \to \mathbb{R}$ and a threshold $t \geq 0$, the* shifted surrogate *$\widehat{f}_{shift} : \mathcal{X} \to \mathbb{R}$ is simply defined by*

$$\widehat{f}_{\mathit{shift}}(x) = \widehat{f}(x) + t \tag{3}$$

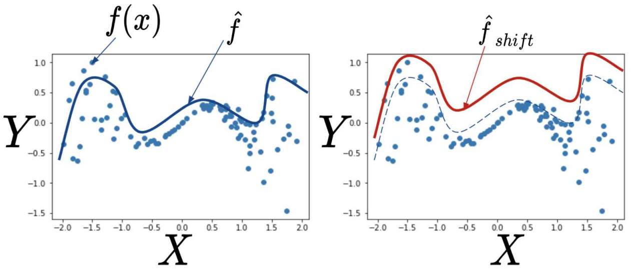


Figure 3: Illustration of the shift operation on a uni-dimensional surrogate. The blue line corresponds to the graph of the surrogate. Since several outputs $f(X_i)$ are under-estimated, we compute the largest non-negative error $t = \max\big(0, \max_i \{f(X_i) - \widehat{f}(X_i)\}\big)$ and shift the surrogate's predictions upwards, by adding t everywhere.

Let $t = \max\big(0, \max_i \{f(X_i) - \widehat{f}(X_i)\}\big)$ denote the largest non-negative error on the test set. When shifting the surrogate's predictions upwards with t, the shifted surrogate never under-estimates the reference model, i.e.,

$$\frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{f(X_i) > \widehat{f}(X_i) + t\}} = 0 \; .$$

It is thus tempting to use Proposition 1 with $\widehat{G}_n = 0$, but this is not allowed since t depends on the test set which is also used to estimate the under-estimation probability. Instead we use a uniform deviation inequality stated in Theorem 1 below. It provides an upper bound $G(t) \leq G^{+}(t)$ similar to that of Proposition 1, but which holds simultaneously for all (possibly data-driven) shifts $t \geq 0$.

Theorem 1 (A uniform Bernstein-type inequality) *Consider $n \geq 2$ independent random variables $X_1, \dots, X_n$ drawn from the same distribution P_X in the domain of study, and independent of the training set ($\widehat{f}$ is considered as fixed). We define $G(t)$ and $\widehat{G}_n(t)$ for all $t \in \mathbb{R}$ by*

$$G(t) = \mathbb{P}(f(X) > \widehat{f}(X) + t)$$

$$\widehat{G}_n(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{f(X_i) > \widehat{f}(X_i) + t\}}$$

Let $\delta \in (0, 1)$. Then, with probability at least $1 - \delta$ over the choice of the test set $X_1, \dots, X_n$, we have: for all $t \in \mathbb{R}$,

$$G(t) \leq \underbrace{\widehat{G}_n(t) + \sqrt{\frac{2\widehat{G}_n(t)}{n} \ln\left(\frac{n}{\delta}\right)} + \frac{5.67}{n} \ln\left(\frac{n}{\delta}\right)}_{=: G^{+}(t)} . \quad (4)$$

The constant 5.67 has not been optimized and could probably be slightly improved. [2] The proof of this well-known inequality follows from several calls to Theorem 2.10 by (Boucheron, Lugosi, and Massart 2013) at quantiles $t = G^{-1}(k/n)$ for $k = 1, 2, \dots, n-1$ (where G^{-1} is a generalized inverse), and from the fact that true and empirical cumulative distribution functions are non-decreasing. The result is valid without any assumption on G.

The above theorem yields the important following corollary, which is valid for any reference model $f : \mathcal{X} \to \mathbb{R}$ and any surrogate model $\widehat{f} : \mathcal{X} \to \mathbb{R}$. The fact that we only guarantee safeness with high probability (upper bound on the under-estimation probability $G(t_{\max})$) instead of 100%-safeness is somehow the price to pay for not requiring any assumption on f nor $\widehat{f}$.

Corollary 1 (Safeness proof for shifted surrogate) *Under the same assumptions as in Theorem 1, denote by $t_{\max}$ the largest non-negative error on the test set:*

$$t_{\max} = \max\Big(0, \max_{1 \leq i \leq n} \big\{f(X_i) - \widehat{f}(X_i)\big\}\Big) .$$

Then, $\widehat{G}_n(t_{\max}) = 0$ so that, with probability at least $1 - \delta$ over the choice of the test set $X_1, \dots, X_n$,

$$G(t_{\max}) \leq \frac{5.67}{n} \ln\left(\frac{n}{\delta}\right) , \quad (5)$$

which is a high-confidence proof (with confidence level $1-\delta$) that the shifted surrogate $\widehat{f}_{shift} = \widehat{f} + t_{\max}$ is $(1-\varepsilon)$-safe, with $\varepsilon = 5.67 \ln(n/\delta)/n$.

We stress that shifting the surrogate impacts its performances as it may increase drastically the gap between the target and its predictions. Hence, encouraging the surrogate to be safe during the training is highly relevant. However, as for the statistical based loss functions described in the Related Works section, there are no existing theoretical results to guarantee that training a surrogate with safeness-promoting constraints is indeed safe in the sense of Definition 1. We thus recommend to:

(i) first build a surrogate with safeness-promoting constraints (so that shifting in Step (ii) does not deteriorate the surrogate too much);

(ii) then shift the surrogate as in Corollary 1 to guarantee safeness with high probability.

In the next section, we design a new loss function to make the training step (i) more safeness-promoting.

Training a Safe Surrogate

In order to minimize the required shift for our surrogate $\widehat{f}$, it is natural to take this shift into account during the learning stage. We design a new loss function that we call *Shifted Mean Squared Error* or SMSE for short. It corresponds to the average squared error between the true target and the shifted predictions on a minibatch. On each minibatch the shift is the largest positive error on that minibatch. A pseudocode describing our loss function is provided in Algorithm 1. The size of the minibatches must be large enough so as to be a good approximator of the shift on the test set.

Algorithm 1 SMSE: *Shifted Mean Squared Error*

```
Require: Target values $y$, predictions $\widehat{y}$, minibatch $\{(x_i, y_i)\}_{i=1}^{n}$
    # approximate the shift that will be operated on the test set by the maximum error on the minibatch
    $t = \max(0, \max_i \{y_i - \widehat{y}_i\})$
    $SMSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \widehat{y}_i - t)^2$
    return SMSE
```

For stability issue, we recommend to pre-train the surrogate with MSE. After this pre-training, the network is modified by stacking on top of it a Dense Layer made of one multiplicative factor plus a bias factor, respectively set to one and zero initially. This allows the surrogate not to be modified in case the shifted pre-trained network would be optimal.

Experiments

We apply our safe learning method (Steps (i) and (ii) in the last section) on three different datasets, with the objective to promote the efficiency of shifted surrogates when trained with safeness constraints and to demonstrate the potential of our method on an industrial use-case. We compare a surrogate trained with the SMSE loss function with three different loss baselines:

1. MSE: *Mean Squared Error*

[2] We could also probably replace the terms $\ln(n/\delta)$ with $c_1 \ln(c_2/\delta)$ for some constants $c_1, c_2 > 0$, using self-normalized empirical process results from (Shorack and Wellner 1986).

2. A-MSE: *Asymmetric Mean Squared Error*.

$$\mathcal{L}_{A\_MSE}(y, \widehat{y}) = \exp\left(\alpha(y - \widehat{y})\right) - \alpha(y - \widehat{y}) - 1$$

3. A-QD: *Assymetric QD*. We modify the loss function proposed in (Pearce et al. 2018) to learn only the upper-bound of their prediction interval. [3]

1D toy model

Our first example mimics a 1-dimensional reference model defined as follows:

$$\forall\ x \in [-2, 2]\ ,\ z = \frac{1.5\sin(\pi x) - 0.3}{0.3}$$
$$f(x) = \frac{z + 2\cos(\pi z)}{6}$$

We designed this reference model to have bumps of varying amplitudes that are challenging for a surrogate. We sampled 1100 samples for both the training and the validation of our network, a two layer network with ReLU activation, optimized with RMSProp on mini-batches of size 32. We compute the shift on an independent test set made of 1000 samples.

Figure 4 shows different results comparing the shifted safe surrogate models learnt with different strategies.

The first graph (top-left) shows the evolution of MSE computed on the four shifted surrogates as a function of the size of the test dataset. Vertical bars indicate standard deviation roughly estimated on five runs. In this graph, we show that in this simple case, our training method combined with the safe shift has the least effect on accuracy.

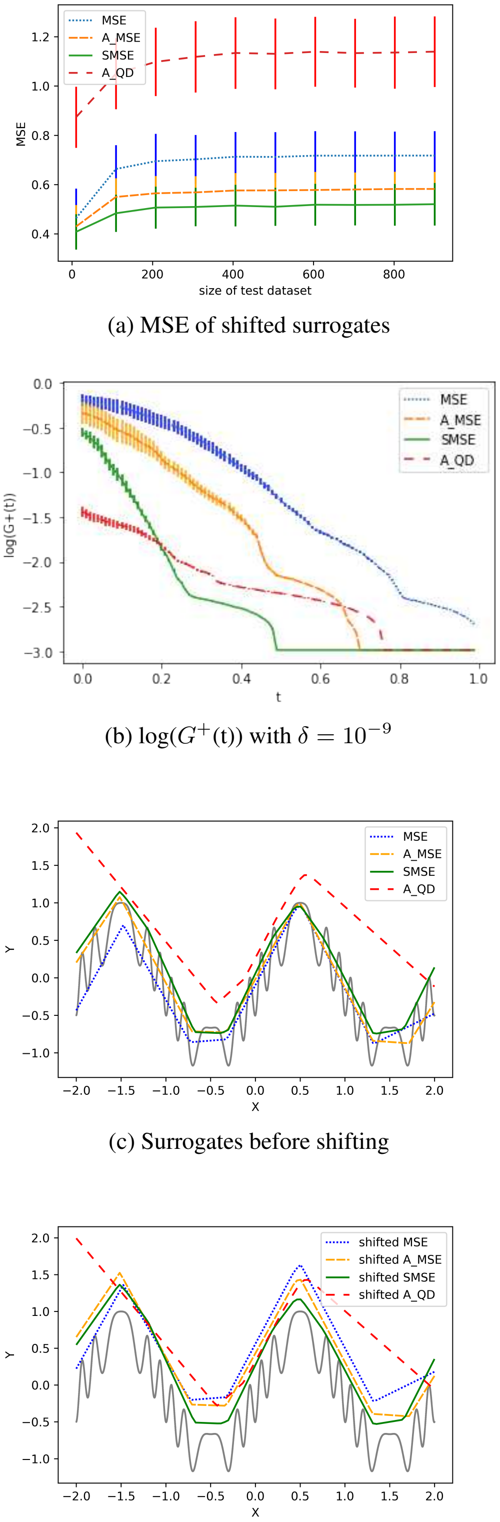
The second graph (top-right) shows the evolution of the upper-bound G^{+}(t) of the under-estimation probability with the shift value t. For $t > 0.2$, the shift SMSE surrogate outperforms all other methods.

The third graph (bottom-left) shows the surrogate models learnt from the 4 different methods. A_QD is mostly overestimating the reference, as the safe MSE method. Standard MSE does not over-estimate, of course, neither does the A-MSE. For this last method, results could be slightly improved by a better tuning of the asymmetric loss function.

The last graph (bottom-right) compares the shifted surrogate models from the 4 different methods with the reference model on its input range. As expected, they all overestimate the reference model but as could be deduced from first graph, SMSE is the most accurate after shifting.

Multidimensional toy model

Since our work focuses on the guarantees of a surrogate on numerical simulations, our second toy dataset focuses on Ordinary Differential Equations (ODEs), specifically, on modeling the propagation of a disease through time, given varying initial conditions. In this lighthearted example, a system of ODEs can be used to model a *zombie-like invasion*, using the equations specified in (Munz et al. 2009). Similarly as



(a) MSE of shifted surrogates

(b) $\log(G^{+}(\mathrm{t}))$ with $\delta = 10^{-9}$

(c) Surrogates before shifting

(d) Surrogates after shifting

Figure 4: 1D toy model use-case

[3] we use the code in the github repository: github.com/TeaPearce/Deep_Learning_Prediction_Intervals

in predictive maintenance, one would need to estimate the number of people affected after a given period, to produce enough treatment. Our training, validation and test set contains respectively 50.000, 10.000 and 10.000 samples.

We train a safe surrogate to over-predict the number of patient after a given amount of time. Our simulation relies on the three basic classes presented in (Munz et al. 2009), plus five ratio per day that describe the behaviors of the population, such as the birth rate. Our regression task consists in predicting the number of patients after five days. Our surrogate is a network of 6 fully connected layers, whose activation functions are either ReLU, either linear. A full description is provided in our code repository.[4]

Figure 5 shows different results comparing the shifted surrogate models learnt with different strategies, averaged on 5 runs.

The first graph (top-left) shows the evolution of MSE computed on the four shifted surrogates as a function of the test database size. The second graph (top-right) discards A_QD method for better comparison. In these graphs, we show that in this case, our training method SMSE combined with the safe shift has the least effect on accuracy. We may need to review our A_QD learning as it seems very poor compared to the other methods.

The third graph (bottom-left) shows the evolution of the upper-bound $G^+(t)$ of the under-estimation probability given the shift value t. For $t > 25$, the shift SMSE surrogate ensures same risk as MSE and A-MSE surrogates.

The last graph (bottom-right) scatters the errors obtained with shift MSE and shift SMSE surrogates given the real number of patient. the errors made by the shifted SMSE surrogates are smaller than for the MSE surrogate for the same associated output.

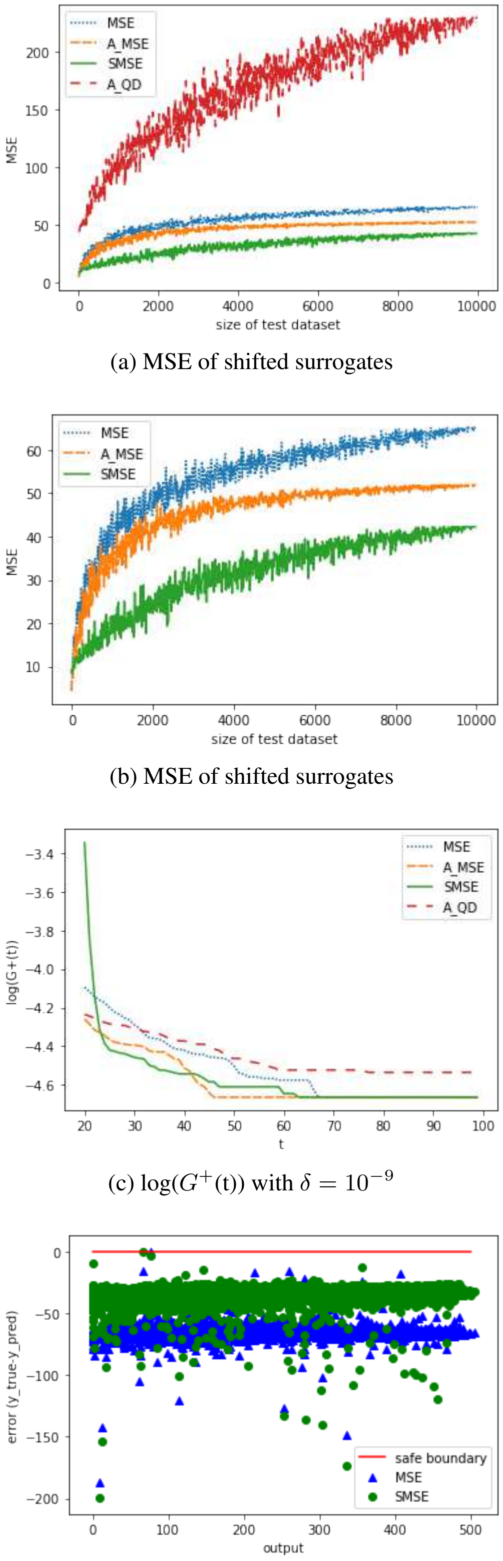
Industrial Use-case

The industrial dataset is sampled from a real simulator used in aeronautics that predicts the landing distance given seven inputs. No additional details can be shared on this use case but the same methods have been applied on it and results are shown in this subsection.

The objective is to train a Neural Network surrogate model that will predict the braking distance with a non-negative error on the whole input domain compared to the reference model. The input domain on which we train the surrogate has been defined by industrial experts to include all potential operational conditions. Our training, validation and test set contains respectively 544.000, 181.000 and 181.000 samples.

Figure 6 shows different results comparing the shifted surrogates learnt with different strategies.

The first graph (top-left) shows the evolution of MSE computed on the four shifted surrogates as a function of the test database size. The second graph (top-right) discards A_QD method for better comparison. In these graphs, we show that in this case, our training method SMSE combined with the safe shift has the least effect on accuracy.



(a) MSE of shifted surrogates

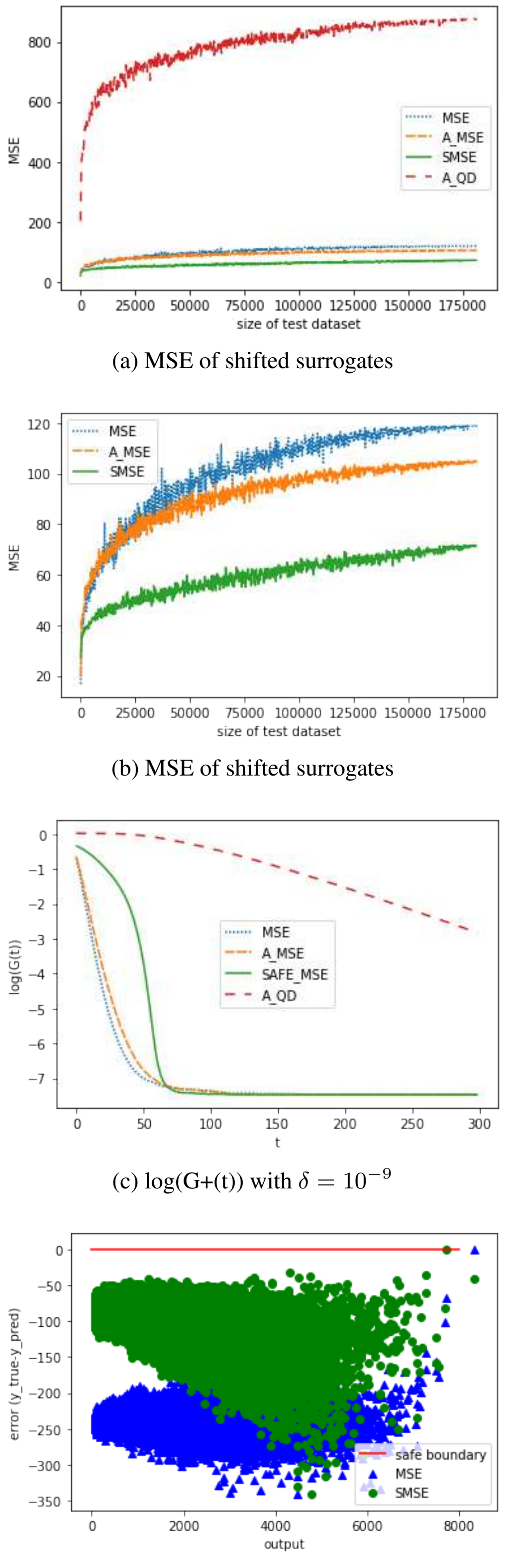
(b) MSE of shifted surrogates

(c) log(G^+(t)) with $\delta = 10^{-9}$

(d) Surrogate accuracy

Figure 5: Zombie use-case

[4] The github adress will be provided after the rebuttal



(a) MSE of shifted surrogates

(b) MSE of shifted surrogates

(c) log(G+(t)) with $\delta = 10^{-9}$

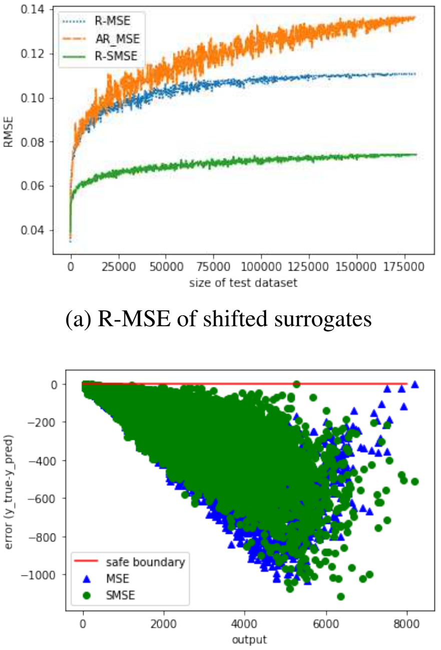
(d) Surrogate accuracy

Figure 6: Braking Distance Estimation use-case

The third graph (bottom-left) shows the evolution of the upper-bound of the under-estimation probability with the shift value t. For $t > 70$, the SMSE surrogate ensures same risk as MSE and A-MSE surrogates.

The last graph (bottom-right) compares the errors of surrogate models built with MSE and SMSE given the landing distance. SMSE shifts are slightly lower than for the MSE surrogate in terms of loss of accuracy.

Since the interval of values for the landing distance is rather large, a relative error seems more appropriate to tackle our problem. Upper-bounding a relative error results into a multiplicative shift instead of an additive shift. Similarly as what has been proposed with SMSE, we can take into account this shift into the training of the loss function. We call this loss function R-SMSE and compare it with the relative pendent of our baselines (R-MSE and AR-MSE). A version of the pseudo-code of R-SMSE is available in Algorithm 2. The efficiency of R-SMSE compared to the baselines is empirically demonstrated in Figure 7.



(a) R-MSE of shifted surrogates

(b) Surrogate accuracy

Figure 7: Braking Distance Estimation use-case trained with relative error

Conclusion

In this paper we studied how to build a safe surrogate model for a given reference model, with the aim of embedding the surrogate into an aeronautical platform. We considered situations where safety means that the surrogate model should over-estimate the reference model, as in the braking distance estimation problem. We used Bernstein-type deviation inequalities to estimate the probability of under-estimating the reference model. This allows to compute a safe shift for any surrogate model. The shift is applied uniformly on all the input domain, so it could deteriorate the accuracy of the surrogate. We thus proposed a way to constrain the surrogate

Algorithm 2 R-SMSE: *Relative Shifted Mean Squared Error*

```
Require: Target values $y$, predictions $\widehat{y}$, minibatch $\{(x_i, y_i)\}_{i=1}^{n}$
  # approximate the relative shift that will be operated on the test set by the maximum error on the minibatch
  $t = \max\left(0, \max_{i, y_i > \widehat{y}_i} \left\{\frac{(y_i - \widehat{y}_i)^2}{\widehat{y}_i^2}\right\}\right)$
  R-SMSE $= \frac{1}{n} \sum_{i=1}^{n} \frac{\left(y_i - (1+\sqrt{t})\widehat{y}_i\right)^2}{((1+\sqrt{t})\widehat{y}_i)^2}$
  return R-SMSE
```

learning phase in order to keep as much accuracy as possible. We compared the newly obtained loss function with other asymmetric loss functions. When all the associated surrogates are post-processed with our safe shift procedure, our loss function yields slightly better accuracy results.

Future work will focus on improving the accuracy of the safe shift. We also plan to compare our approach and guarantees with the very recent work on conformalized quantile regression (Romano, Patterson, and Candès 2019).

Acknowledgements

This project received funding from the French "Investing for the Future – PIA3" program within the Artificial and Natural Intelligence Toulouse Institute (ANITI). The authors gratefully acknowledge the support of the DEEL project.[5]

References

Bernstein, S. 1924. On a modification of chebyshev's inequality and of the error formula of laplace. *Ann. Sci. Inst. Sav. Ukraine, Sect. Math* 1(4):38–49.

Biannic, J.; Hardier, G.; Roos, C.; Seren, C.; and Verdier, L. 2016. Surrogate models for aircraft flight control: some off-line and embedded applications. *AerospaceLab Journal*.

Boucheron, S.; Lugosi, G.; and Massart, P. 2013. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press.

De Rocquigny, E.; Devictor, N.; Tarantola, S.; Lefebvre, Y.; Perot, N.; Castaings, W.; Mangeant, F.; Schwob, C.; Bolado Lavin, R.; Massé, J.-R.; Limbourg, P.; and Kanning, W. 2008. *Uncertainty in Industrial Practice*. John Wiley & Sons, Ltd.

Der Kiureghian, A., and Ditlevsen, O. 2009. Aleatory or epistemic? does it matter? *Structural Safety* 31(2):105–112.

FAA, F. A. A. 2016. Verification of adaptive systems. http://https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/media/TC-16-4.pdf.

Frogner, C.; Zhang, C.; Mobahi, H.; Araya, M.; and Poggio, T. A. 2015. Learning with a wasserstein loss. In *Advances in Neural Information Processing Systems*, 2053–2061.

Gal, Y., and Ghahramani, Z. 2015. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv:1506.02158*.

Hilderman, V. 2014. Understanding do-178c software certification: Benefits versus costs. In *2014 IEEE International Symposium on Software Reliability Engineering Workshops*, 114–114. IEEE.

Jian, Z.-D.; Chang, H.-J.; Hsu, T.-s.; and Wang, D.-W. 2017. Learning from simulated world-surrogates construction with deep neural network. In *SIMULTECH*, 83–92.

Keren, G.; Cummins, N.; and Schuller, B. 2018. Calibrated prediction intervals for neural network regressors. *IEEE Access* 6:54033–54041.

Khosravi, A.; Nahavandi, S.; Creighton, D.; and Atiya, A. F. 2010. Lower upper bound estimation method for construction of neural network-based prediction intervals. *IEEE transactions on neural networks* 22(3):337–346.

Lathuilière, S.; Mesejo, P.; Alameda-Pineda, X.; and Horaud, R. 2019. A comprehensive analysis of deep regression. *IEEE transactions on pattern analysis and machine intelligence*.

Munz, P.; Hudea, I.; Imad, J.; and Smith, R. J. 2009. When zombies attack!: mathematical modelling of an outbreak of zombie infection. *Infectious disease modelling research progress* 4:133–150.

Papernot, N.; McDaniel, P.; Wu, X.; Jha, S.; and Swami, A. 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, 582–597. IEEE.

Pearce, T.; Zaki, M.; Brintrup, A.; and Neely, A. 2018. High-quality prediction intervals for deep learning: a distribution-free, ensembled approach. *arXiv:1802.07167*.

Romano, Y.; Patterson, E.; and Candès, E. J. 2019. Conformalized quantile regression. In *Advances in Neural Information Processing Systems (NeurIPS 2019)*, volume 32.

Shorack, G. R., and Wellner, J. A. 1986. *Empirical Processes with Applications to Statistics*. John Wiley & Sons.

Sudakov, O.; Koroteev, D.; Belozerov, B.; and Burnaev, E. 2019. Artificial neural network surrogate modeling of oil reservoir: a case study. In *International Symposium on Neural Networks*, 232–241. Springer.

Tagasovska, N., and Lopez-Paz, D. 2019. Single-model uncertainties for deep learning. In *Advances in Neural Information Processing Systems 32*. 6414–6425.

Tolstikov, A.; Janssen, F.; and Fürnkranz, J. 2016. Evaluation of different regression learners under asymmetric loss for predictive maintenance. Technical Report TUD-KE-2015-02, Technische Universität Darmstadt.

Wilhelm, R.; Engblom, J.; Ermedahl, A.; Holsti, N.; Thesing, S.; Whalley, D.; Bernat, G.; Ferdinand, C.; Heckmann, R.; Mitra, T.; Mueller, F.; Puaut, I.; Puschner, P.; Staschulat, J.; and Stenström, P. 2008. The worst-case execution-time problem&mdash;overview of methods and survey of tools. *ACM Trans. Embed. Comput. Syst.* 7(3):36:1–36:53.

Yao, Q., and Tong, H. 1996. Asymmetric least squares regression estimation: a nonparametric approach. *Journal of nonparametric statistics* 6(2-3):273–292.

[5] https://www.deel.ai/