EC4.31

Methodological Guideline for Conformal Prediction with PUNCC

Document reference number for ANR

contact@confiance-ai.fr | www.confiance.ai

Document reference: 431B

Contributors

	Name	Organisation
Responsible for the deliverable	Mouhcine Mendil	IRT Saint Exupéry
Co-authors	Luca MOSSINA	IRT Saint Exupéry
	Samuel Kierszbaum	Airbus Protect
Reviewers	Corentin Friedrich	IRT Saint Exupéry

Document Control

Revision	Date	Commentary	Author
v1.0	29/11/2023	Document creation	Mouhcine Mendil
v1.1	18/12/2013	Review start	Mouhcine Mendil
v2.0	22/12/2013	Reviewed version	Mouhcine Mendil

Contents

A. Introduction and abstract

A.1. General introduction to trustworthy AI challenges

Trustworthiness in AI within critical systems (systems that can directly or indirectly affect human life and moral entities) is essential for its widespread adoption (by the industry, the decision makers, the general public, etc.) and poses the following significant challenges.

- First, how to design AI models, so that, by construction, they satisfy trustworthy properties (accuracy, robustness. . .).
- Secondly, how to characterize these AI models, for example to understand and explain their behavior and their adequacy to the operational domain.
- Then, how to implement and embed those AI models on hardware, by making them fit for the target without losing their trustworthy properties.
- Another question is, what methods of data engineering to apply in order to, among other topics, manage important volumes of data and adapt to the evolution of the operational domain.
- At system level, what verification and certification processes to consider specifically for AI-based systems.
- Finally, a federation of all these matters is necessary to build an end-to-end methodological approach, supported by a consistent engineering environment compatible with industrial practices.

These are the challenges, among others, that the Confiance.ai program addresses.

A.2. Context and Scientific Challenges

Widespread adoption of *Machine Learning* (ML) models raises concerns about their reliability and robustness. *Uncertainty Quantification* (UQ) is a crucial aspect of ML that aims to quantify the uncertainty associated with model predictions. This is particularly important in high-stakes applications, such as in health, transportation and defense, where making erroneous predictions can have severe consequences. Traditional UQ approaches, such as confidence intervals and Bayesian networks, rely on assumptions about the underlying data distribution. While these methods are commonly used, their robustness is compromised by the violations of these assumptions, which could lead to unreliable uncertainty estimates.
Thus, we present this methodological guideline, which is derived from the project EC4 of confiance.ai, also named "Design for Trustworthy AI: algo, module and systems levels". More specifically, the goal of this guideline is to present approaches and tools that have been developed to address the challenge of rigorous UQ in different domains.

Conformal Prediction (CP) is a key approach that can replace or complement traditional UQ approaches. CP offers distribution-free guarantees, meaning that its validity does not depend on specific assumptions about the data distribution. It is also model agnostic, i.e. it can be applied to a wide range of ML models without the need for model-specific assumptions or modifications. This makes CP a more robust and reliable approach to UQ, particularly in real-world scenarios

where data distributions may be complex or unknown. Finally, by being a post-hoc approach, CP enables non-intrusive integration and seamless introduction of uncertainty quantification into existing ML pipelines without requiring extensive modifications to existing models or workflows.

The main confiance.ai scientific challenges addressed in this work are summed as follows:

- **Identify biases in data/knowledge leading to poor decisions**
- **Calculation of the confidence score by various approaches**
- **Methodology for identifying risks in the case of trusted AI systems**

A.3. Rationale and Document Organization

This document serves as an entry point to the practical adaptations and applications of CP using **puncc** library in confiance.ai. It guides the reader through diverse scenarios that require robust UQ and provides comprehensive context along with potential results to be expected upon correct use of the proposed approaches, seamlessly integrated in **puncc**. The document is organized as follows:

- Chapter A serves as an introduction and a roadmap to this methodological guideline, with a showcase of the context, rationale, scientific objectives, organization, target audience and usage.
- Three Chapters B, C and D, each of which present the implementation of CP in a specific domain. Such chapters aim to introduce the corresponding methodology, including its overall vision, relevant context, unique contributions, prerequisites, level of maturity, explanatory diagrams and perspectives.
- Finally, we conclude this document with Chapter E that recapitulates the main contributions and limitations of our approaches.

A.4. Target Audience and How to Use This Document

This document is primarily intended for data scientists, ML practitioners and any user seeking to leverage conformal prediction to rigorously quantify the uncertainty of their models and to explore the various applications of the library **puncc** in confiance.ai. The document provides a comprehensive understanding of the rationale behind each application, enabling users to evaluate potential utility and limitations based on their specific use case and needs.

B. Conformal prediction and NLP

This chapter describes the work done in EC4N31 regarding the use of the *puncc* library for uncertainty quantification in the context of a document classification problem with textual data from the use-case "UC_Renault_Opinion_Mining".

B.1. Description of the method

Puncc currently supports two conformal algorithms that can be used in the context of a document classification problem: RAPS and APS.
Both are techniques for constructing prediction sets (i.e. a set of classes related to the classification problem at hand) with formal coverage guarantees.
This guarantee ensures that when a model utilizes one of these algorithms to generate a prediction set for a given instance, there is a fixed probability, determined by the user through the parameter α, that the true class belongs to that set with $[100 \times (1-\alpha)]\%$ certainty.
RAPS represents an enhancement over APS as it introduces a regularization technique. This improvement results in smaller prediction sets on average while still maintaining the specified coverage.

B.2. Context

With respect to the End-to-End Approach, this guideline fits into the context "Complementary software items for UQ: ML Model encapsulated in Conformal Predictor".

B.3. Procedure

To harness *puncc* for robust uncertainty estimations in a Natural Language Processing (NLP) classifier for a classification task, one must undertake four essential steps: choose a model tailored for the classification task, compile a set of annotated data, select a value for α, and opt for a conformal classification algorithm from *puncc* such as RAPS or APS. Figure B.1 illustrates how this procedure is applied in the case of the document classification problem from the use-case "UC_Renault_Opinion_Mining".
In this scenario, a distilled RoBERTa model serves as the NLP classifier, chosen for its compact size and high performance. The conformal prediction algorithm RAPS is employed, along with the use-case-specific data and an α value set at 0.1 (10%).
The annotated data is partitioned into a training dataset, a calibration dataset, and a test dataset. Following the model training on the training data, a calibration process is performed using the calibration dataset. Subsequently, the calibrated model is applied to conduct conformal classification on the test dataset. Each conformal prediction yields a prediction set comprising a list of plausible classes. Overall, it is observed that approximately 90% or $100 \times (1-\alpha)$ of the prediction sets correctly encompass the true class.

B.4. Implementation

One can find an implementation of the conformal uncertainty quantification procedure applied to the textual data from the Renault use-case "UC_Renault_Opinion_Mining" here[1].

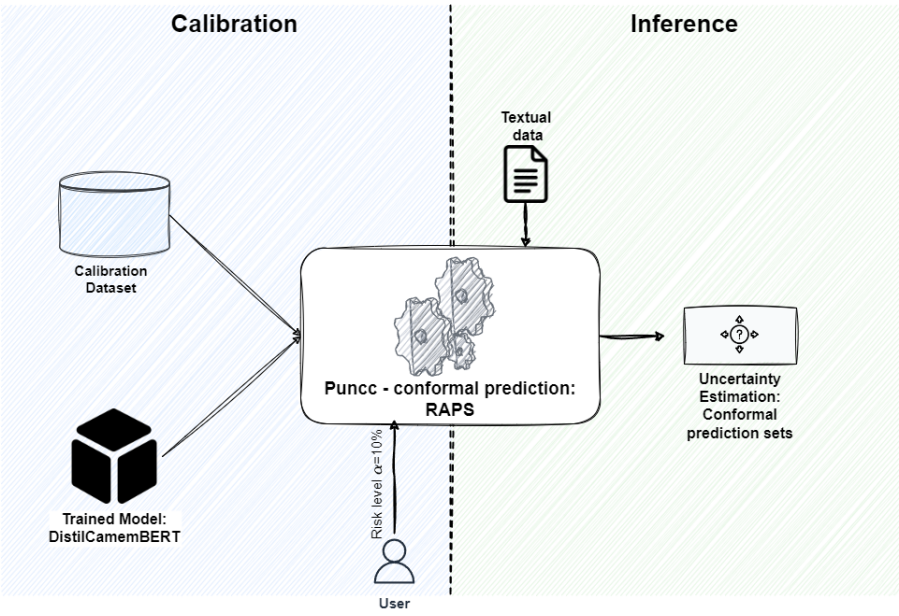

Figure B.1: Conformal uncertainty quantification procedure on Renault data

B.5. Maturity

On batch 3, the component **puncc** is aiming for functional and technical maturity levels 3.

[1] https://git.irt-systemx.fr/confianceai/Demonstrators/-/tree/master/Comp_350_2?ref_type=heads

C. Conformal prediction and image segmentation

This chapter describes some **work in progress** on the application of *Conformal Prediction* (CP)[1] (Vovk et al., 2005; Angelopoulos and Bates, 2021) to *Semantic Image Segmentation* (SIS) (Mo et al., 2022).

As of the end of Batch 3 (Q4 2023), these functionalities have not yet been implemented in *Predictive UNcertainty Calibration and Conformalization* (PUNCC), although they are part of our proposal for Batch 4 (Q1-Q3 2024). Here we give a **methodological preview** of what is to come, with details on semantic segmentation, what UQ is in that context, how to implement CP and how interpret it.

C.1. Context

With respect to the End-to-End Approach, this guideline fits into the context "Complementary software items for UQ: ML Model encapsulated in Conformal Predictor".

C.2. Semantic Segmentation.

Semantic image segmentation deals with associating each pixel of an input image to a class among $C = \{c_1, c_2, \ldots, c_K\}$: for instance, one could have $C = \{c_1 = \text{pedestrian}, c_2 = \text{car}, \text{etc.}\}$.

Segmentation models are trained on labeled data $D_{\text{train}} = \{X_i, Y_i\}_{i=1}^{n}$: $\widehat{f}$ is a **pretrained** model mapping an input image with X to a predicted segmentation mask $\widehat{Y}$.
We call X an input image H pixels high and W pixels wide, totalling $n_{\text{pixels}} = H \times W$ pixels. We call Y the "ground-truth" semantic segmentation mask annotated by an expert, available for training and conformalizing samples. We assume Y and the prediction $\widehat{Y} = \widehat{f}(X)$ to be arrays of the same size as the input image, containing n_{pixels} pixels.

For each pixel $\hat{y}_{ij} \in \widehat{Y}$, we also assume to have the softmax scores associated to each of the $|C| = K$ possible classes, such that we have:

$$\sigma(X_{ij}) = [\sigma^{c_1}(X_{ij}), \sigma^{c_2}(X_{ij}), \ldots] \tag{C.1}$$

$$\sum_{c_j \in c_1 \ldots c_K} \sigma^{c_j}(X_{ij}) = 1. \tag{C.2}$$

In Figure C.1 is an example of labelization, with the input image X on the left, and the annotated version on the right (background class in dark gray).

C.2.1 Uncertainties in semantic segmentation

As an example of uncertainty in SIS, in Figure C.2 we can see several patches of the input image which were mislabeled by the predictor. These patches contain critical elements, notably

[1] Readers familiar with CP can skip Section C.3. CP has been already introduced and discussed in the reports of batch 1 (EC3.5) and batch 2 (EC4.20). For more details, see their corresponding chapters and references. Alternatively, Section C.3 provides a brief, self-contained introduction to these topics.

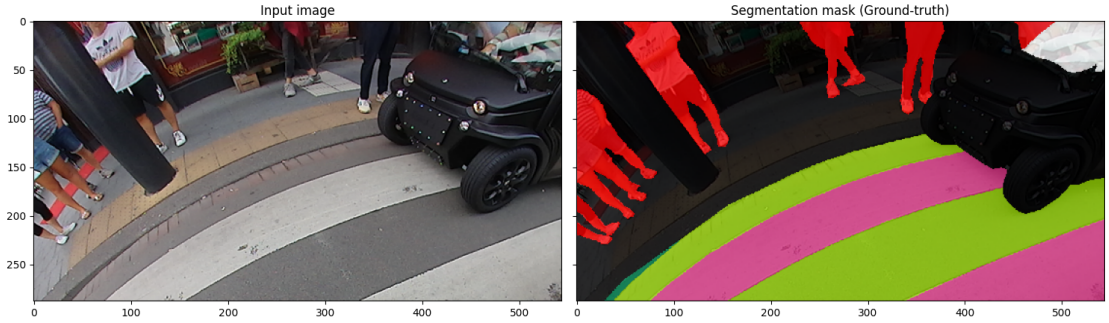

Figure C.1: An example of input image X (on the left) and the expertly annotated ground-truth Y semantic mask (on the right). In red are all the pixels labeled as "person".

pedestrians and bike riders. In the lower right-hand-side figure, we plot, pixel by pixel, the difference between the score of the most highly valued class (the "top 1" class) and the score of the true class: if the prediction was correct, than this difference is zero. On the other hand, if this difference is close to one (darkest shade of red in the figure), than the true class received (erroneously) a very low score and hence is very "far" from being in the (correct) first position. We can say that the model $\widehat{f}$ has made a severe predictive mistake for those pixels.

We can draw some conclusions. Unsurprisingly, the edges are subject to higher errors: partly due to pixelation of images, partly due to labeling imprecisions (could be unavoidable), these errors are relatively benign, as they constitute a small percentage of the segmented class. More worryingly, we can see **whole patches** in dark red in the lower right-hand-side picture: for some classes in the picture, all pixels were mislabeled. It could be operationally unacceptable if this error were to incur in the detection of pedestrians or cyclists.

C.3. Conformal prediction.

This section only aims at explaining the idea of CP and its interpretation. For more details, including the definition of *nonconformity scores* and the conformalization algorithms, see the reports of batch 1 (EC3.5) and batch 2 (EC4.20).

Conformal Prediction (CP)[2] is an approach to UQ based on prediction sets: normally, when we run a ML predictor we obtain a prediction $\widehat{Y} = \widehat{f}(X)$, which is known as a "point prediction" (e.g. a scalar in regression, a class in classification). This prediction, however, does not carry any information on the uncertainty associated to prediction: can we trust the model or the data?

The idea of CP is to build a theoretically valid set that contains multiple likely values, not just the one with the highest score. This is common in regression, in the form of a prediction interval, but probably not for classification. See the following simplified examples to get a practical idea:

$$\widehat{Y}^{\text{reg}} = 23.22 \longrightarrow \widehat{C}(X) = [\widehat{Y} - 1.74, \widehat{Y} + 1.74] = [21.44, 24.96];$$

> *"For our itineraries, we usually consume about 23 liters of diesel, with an error of around two liters".*

[2]This section can be skipped. See footnote 1

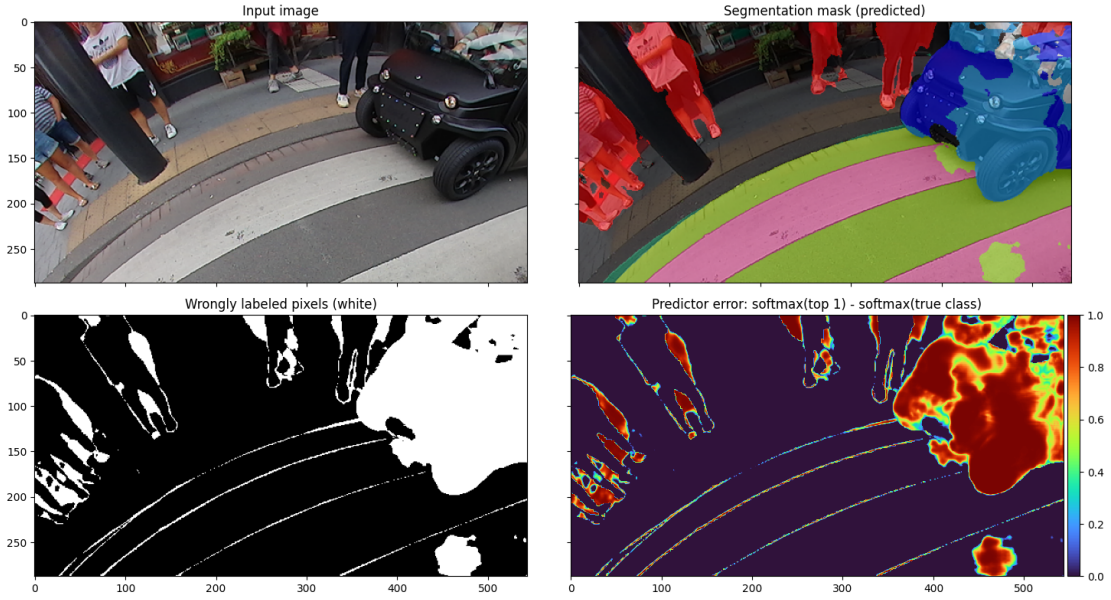

Figure C.2: Example of prediction: input image, output prediction mask, wrong predictions (white pixels), prediction error. The quantified prediction error is the difference between the highest score (predicted class) and the score of the true class: $\text{err} = \sigma(x_{ij})_{(1)} - \sigma(x_{ij})_{\text{true class}}$, for pixel (i, j).

$$\widehat{Y}^{\text{classif}} = \{\text{pedestrian}\} \longrightarrow \widehat{C}(X) = \{\text{pedestrian, rider, tree}\}. \tag{C.3}$$

"This picture looks is recognized as being a pedestrian, but it could also be a person on a bicycle, given what is visible".

These statements can be made more rigorous with CP, via the following probabilistic construct:

$$\mathbb{P}\{Y \in C(X)\} \geq 1 - \alpha. \tag{C.4}$$

"Following the procedure of Conformal Prediction many times, on average, I will get prediction sets that contain the ground truth with probability $1 - \alpha$.
If I set my tolerable risk at 10%, than 90% *of the time the prediction sets contain the correct label. "*

In words, on average over many repetitions of the CP method, the prediction set $C(X)$ is guaranteed to contain the ground truth Y with probability at least $1 - \alpha$.

Our propensity to accept a predictive error is given by the (small) number α, e.g. $\alpha = 0.1$ or $\alpha = 0.01$, chosen by the users; this is also known as *miscoverage rate*, or risk. There is "no right way" to select this: just like with traditional hypothesis testing, one must first formulate their problem ("I cannot have more than one error per 100 predictions") and then proceed to collect the *calibration data* D_{cal} and build conformal predictor.

Crucially, CP is a distribution-free, model-agnostic method: it is applicable to any predictor with no assumptions on the distribution of the data, other than having i.i.d. calibration data. However, this comes at a cost: the guarantee is said to be *marginal* over (X, Y), that is, the guarantee is *on average* over the distribution of the data. For more details, see the hands-on

tutorial by Angelopoulos and Bates (2021), which also presents some methods to circumvent this limit[3].

Conformalization in a nutshell We mention only the procedure of *Inductive* CP (Papadopoulos et al., 2002), which has become the standard approach.

First, we need to sample some *conformalization* (or *"calibration"*) data $D_{\text{cal}} = (X_i, Y_i)_i^{n_{\text{cal}}}$ from the same distribution as the inference data: that is, we should sample from the real-world data distribution that we will encounter during our inferences. Then, we define a *nonconformity score* $S(X,Y)$, a kind of residual, that embeds our *notion of error*.

For classification, we follow the Least Ambiguous Classification sets (LAC) of Mauricio Sadinle and Wasserman (2019).
Let y' be the true class of input X. For a calibration point (X,Y) we find the smallest threshold δ such that:

$$S(X,Y) = \inf\{\delta : \sigma^{y'}(X) \geq \delta\}. \tag{C.5}$$

The meaning of the score in Eq. C.5 is clear when we define the **prediction set** $C(X)$:

$$C_\delta(X) = \{y \in \mathscr{Y} : \sigma^y(X) \geq \delta_i\} \tag{C.6}$$
$$\cup \{y : \sigma^y(X) \text{ is the highest scored class}\}. \tag{C.7}$$

Eq. C.6 tells us to include all classes whose softmax score is higher than δ_i. The second part, in Eq. C.7, ensures to always include the class with the highest score to *avoid building empty prediction sets*.

C.4. Conformal semantic segmentation

In Section C.3 we have seen how to build prediction sets for the case of classification; for SIS, one inference corresponds to repeating this procedure for the $n_{\text{pixels}} = W \times H$ pixels in the input image and yielding n_{pixels} prediction sets.

Nonconformity score. We apply the formulation of prediction sets of Eq. C.6 and extend it to segmentation with a slight modification: we want to cover at least $\tau \times 100\%$ of the pixels, since covering 100% of pixels is too strong a requirement. The **minimum coverage** parameter $\tau_{\text{cov}} \in [0,1]$ is set by the user, to a value such as 95%, for instance.
For every ij-th pixel in the image, we apply Eq. C.6 as $S(X_{ij}, Y_{ij})$, so that we compute the highest softmax threshold according to which we would have included the true class of the pixel. In this computation, we disregard the geometry of the pixels and take them to be a simple sequence (i.e. a "flattened array") of pixel-wise scores:

$$R(X,Y) = \left\{S(X_{ij}, Y_{ij})\right\}_{i,j} := (\delta_k)_{k=1}^{n_{\text{pixels}}} \tag{C.8}$$

For a calibration image (X_i, Y_i) the score is then:

$$S(X_i, Y_i) = \lceil \tau_{\text{cov}} \times n_{\text{pixels}} \rceil - th \text{ element of } R(X_i, Y_i) = \delta^i_{(\lceil \tau \times n_{\text{pixels}} \rceil)} \tag{C.9}$$

[3] In practice we want *conditional coverage*, that is, a guarantee for every possible $X = x$. This is not possible without making strong assumptions on the data and the predictor.

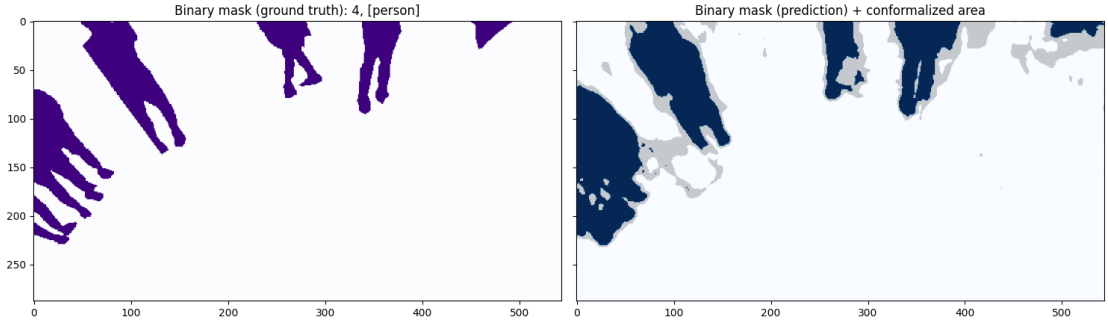

Figure C.3: Detail of an inference: of all the ten classes, we show the effect of conformalization on the class "person". On the left (purple) is the ground truth for that class, on the right blue, all the pixels that are classified as "person". In a light shade of (gray), is the "augmented" region of pixels included after conformalization. If this algorithm were to be applied in a complex pipeline, to be confident at the $(1-\alpha)$ to cover τ_{cov} pixels, we would need to include the shaded area in our algorithm, resulting in a potentially more conservative behaviour.

Now, we repeat the computation for all calibration points and collect the scores in $\bar{S}$.

$$\bar{S} = \left(\delta^i_{(\lceil \tau_{\text{cov}} \times n_{\text{pixels}} \rceil)}\right)_{i=1}^{n_{\text{cal}}}. \tag{C.10}$$

Conformalizing quantile. For a confidence level $(1-\alpha)$ fixed by the user[4], e.g. 97%, we compute the *conformalizing quantile*:

$$q_{1-\alpha} = \lceil (1-\alpha) \times (n_{\text{cal}}+1) \rceil - th \text{ largest element of } \bar{S}. \tag{C.11}$$

Building the prediction set. Finally, we extend the classification prediction set in Eq. C.6 as follows:

$$C_\delta(X) = \{y_{ij} \in \mathscr{Y} : \sigma^{y_{ij}}(X) \geq q_{1-\alpha}\} \tag{C.12}$$

$$\cup \{y : \sigma^{y_{ij}}(X) \text{ is the highest scored class}\}. \tag{C.13}$$

$$\forall (i,j) \in [1 \ldots H] \times [1 \ldots W] \tag{C.14}$$

The above reads as: for every (i,j)-th pixel in $\widehat{Y}$, we output *all* the classes whose score surpasses the conformal threshold $q_{1-\alpha}$ and we say that these classes were "activated". Many activations in a pixel indicate that there was not a single highly confident class (e.g. $\sigma > 0.999$) but the scores were more spread, and the CP procedure (which depends both on the data *and* the pretrained predictor) resulted in more classes in the set, signaling uncertainty.
This is especially useful since, empirically, good predictors exhibit a good behaviour where their softmax are also partially *informative*: they give smaller scores to more "uncertain" pixels, for instance on the borders of predicted mask.

[4]The nominal miscoverage α is completely independent of the minimum coverage τ_{cov}

C.5. Conclusion and Perspectives

This application of CP has yielded promising results: this form of UQ does not rely on any hypothesis on the predictor and can be applied to existing architecture almost for free. This can work as a good, theoretically-grounded first step in building trustworthy segmentation models, to be integrable in more complex AI pipelines for critical systems. Finally, we are planning on extending this and other existing state-of-the-art methods in conformal segmentation in PUNCC, with a minimal burden on the user and facilitating adoption of these kind of methods.

D. Conformal Anomaly Detection

This is a brief summary of the work done in EC5N25. For more details, we refer the reader to the official methodological guidelines for *Conformal Anomaly Detection* (CAD), found in the document entitled "Methodological Guideline for Time Series Anomaly Detection".

D.1. Description of the method

Conformal Anomaly Detection (CAD) is a statistical approach for calibrating anomaly detectors that learn from data. The benefit of such method is to provide theoretical guarantees for controlling the *False Detection Rate* (FDR) within a level of risk α (e.g. 5%) chosen by the user. In other words, CAD enables to upper-bound false alarms with a rate selected by the operator. In practice, CAD relies on CP to allow users to set a threshold for how many false positives they are willing to tolerate. As a result, the anomaly score threshold is automatically adjusted to guarantee that the FDR does not exceed that limit. This particular attribute holds significant importance within the industrial context as it effectively mitigates the burden of incessant false alarms, thereby conserving the time and energy of system operators at a minimal cost. Also, it is very useful for unsupervised anomaly detection, as it eliminates the need for labeled data or expert knowledge.

D.2. Context

With respect to the End-to-End Approach, this guideline fits into the two contexts, depending on how the CAD framework is used: "Complementary software items for UQ: ML Model encapsulated in Conformal Predictor".

D.3. Procedure

CAD hinges on a lightweight post-processing step, which is both computationally efficient and imposes no additional requirements other than the availability of a holdout dataset. Such approach can be used with any anomaly detector that produces an anomaly score $\hat{s}(x)$ for each data point x. Additionally, we need a new dataset of n samples, called the calibration dataset, which will be used to evaluate the calibrated anomaly threshold. The calibration dataset should not have been used during the training phase. The overall procedure is illustrated in Figure D.1.

D.4. Implementation

- CAD is implemented in the open-source library **puncc**[1] under the anomaly detection module. Comprehensive documentation is available online.
- The application of CAD on benchmarks (numenta[2] and toy datasets) and on the EMS use-case are available at `https://git.irt-systemx.fr/confianceai/ec_5/ec5_as3/`

[1] `https://github.com/deel-ai/puncc`.
[2] `https://www.numenta.com/resources/htm/numenta-anomoly-benchmark/`

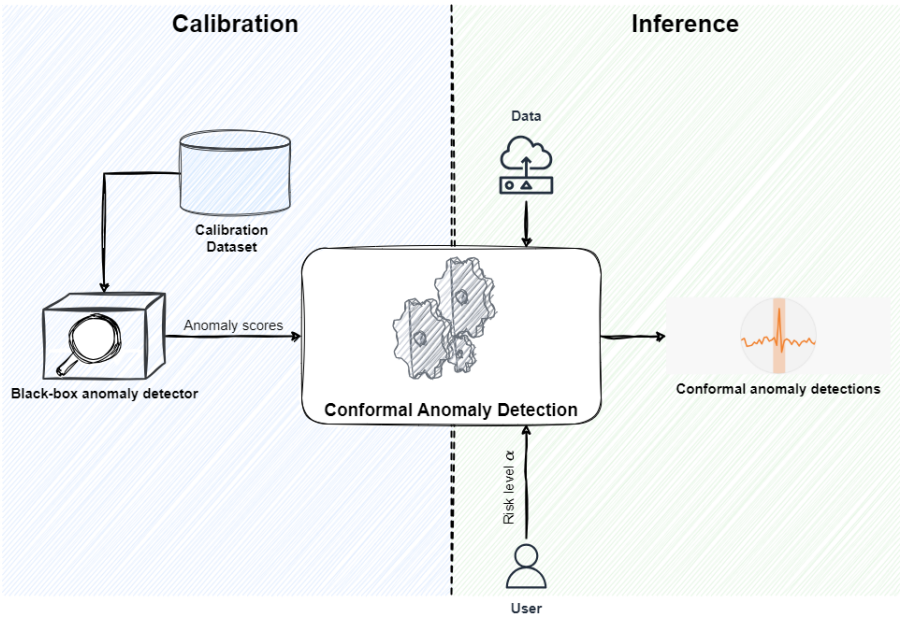

Figure D.1: Conformal Anomaly Detection Procedure

cad.

D.5. Maturity

On batch 3, the component **puncc** is aiming for functional and technical maturity levels 3.

D.6. Guarantees

The choice of calibration dataset is of utmost importance in CAD. The FDR control guarantee for anomaly detection is valid when the calibration examples and the test point are i.i.d. In practice, it is sufficient for the anomaly scores estimated on the calibration and the test sets to be i.i.d to guarantee the FDR control.

There are two explanations for the cause of a conformal anomaly: 1) it may correspond to a rare or previously unseen, yet normal, example that happens with probability at most α, i.e. a false alarm 2) If not, it is a true anomaly in the sense that it was not generated according to the same distribution as the calibration data.

In general, the calibration dataset should include only a small number of anomalous data points (anomaly ratio in the calibration dataset should be at most α). In this setting, the guarantee of controlling the FDR holds in practice. It is noteworthy to mention a rule of thumb for selecting the calibration dataset size n. Roughly speaking, Angelopoulos and Bates (2021) recommends choosing a calibration set of size n = 1000 is sufficient for most purposes. It is also important to note that the calibration dataset should not be used to train the anomaly detector. Otherwise, the anomaly detector will be overfitted to the calibration dataset and will not be able to generalize to new data accurately.

D.7. Contributions

- Implementation of CAD within the open-source library **puncc**.
- Application of CAD on top of classic anomaly detection algorithms on benchmark datasets.
- Application of CAD on top of classic anomaly detection algorithms for EMS data.
- Use of CAD to calibrate the anomaly scores produced by an anomaly detector developed in confiance.ai for EMS data. This allows to control the false discovery rate (FDR) of the anomaly detector.
- General guidelines about CAD and insights into how to use it effectively for EMS data.

D.8. Demonstrator

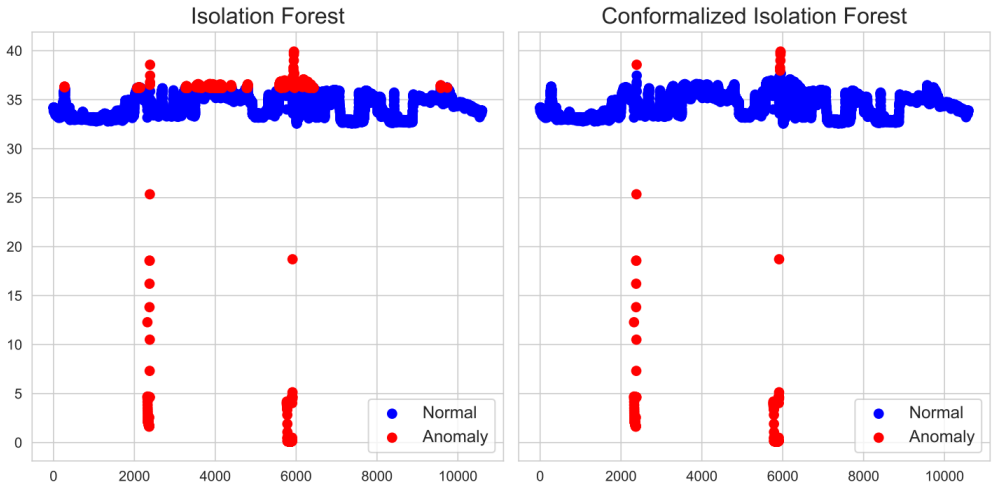

Figure D.2: Results of using CAD on an unsupervised anomaly detector. The isolation forest algorithm is trained then applied on a time series derived from a sensor within the UC Air Liquid EMS. The rate of alarms drops from 9% to 2% when calibrating the anomaly detection threshold using CAD.

The demonstrator compiles CAD applications on academic benchmarks and the *Efficiency Monitoring System* (EMS) use-case. Specifically for EMS, we illustrate the calibration of several anomaly detection models on times series derived from factory sensors. Figure D.2 shows the example of conformalizing an isolation forest to reduce its false anomaly rate.
The following code snippets shows how to easily wrap a blackbox anomaly detector using puncc and run the calibrated model. CAD enables to control the false alarm rate up to the user-defined value of 5%:

```
import numpy as np
from deel.puncc.anomaly_detection import SplitCAD

# Instantiate CAD on top of a pretrained anomaly detector
cad = SplitCAD(anomaly_predictor, train=False)

# Calibrate the threshold using a calibration dataset
cad.fit(z_calib=calib_dataset)
```

```
# We set the maximum false detection rate to 5%
alpha =0.05

# The method ‘predict‘ is called on the new data points
# to predict which are anomalous and which are not
cad_results =cad.predict(new_dataset, alpha=alpha)
cad_anomalies =new_dataset[cad_results]
cad_not_anomalies =new_dataset[np.invert(cad_results)]
```

D.9. Perspectives

The primary advantage of employing CAD lies in its ability to fine-tune the anomaly score threshold, allowing for precise control over the *False Detection Rate* (FDR) up to a user-specified threshold α. Nonetheless, it is essential to recognize that an exclusive focus on minimizing the FDR may inadvertently lead to a different problem: an increase in false negatives. Failing to detect an anomaly can result in substantial financial losses or other forms of damage. Consequently, it is essential for future research to focus on achieving a balance between minimizing false alarms and reducing false negatives, aiming to strike a Pareto-efficient trade-off that optimizes both aspects.

E. Conclusion

Conformal Prediction (CP) is a simple approach that provides reliable uncertainty estimates without relying on specific distributional assumptions. Puncc Python library offers a comprehensive set of tools for implementing CP algorithms and applying them to a wide range of ML tasks. Through its seamless integration with existing ML frameworks and its extensive documentation, puncc empowers developers to incorporate CP into their workflows. This capability enables them to quantify and communicate uncertainty in their models, leading to more informed decision-making and enhanced trust in ML-driven applications.

In the context of NLP, puncc was employed to quantify the uncertainty associated with opinion analysis. In computer vision, puncc was applied to quantify the uncertainty for image segmentation. In time series, puncc was applied on top of anomaly detectors to control false alarms. The results can be utilized to assess the reliability of predictions and inform subsequent actions, such as identifying regions of high uncertainty in images, flagging inaccurate text analysis for human review and mitigating the burden of incessant false anomaly alarms.

The benefits of CP extend beyond academia and research, offering significant value to the industrial sector. By quantifying uncertainty, industrials can enhance the reliability of ML-powered systems and therefore improve decision-making processes. The library's ease of use, coupled with the benefits of CP, makes it a valuable asset for both academia and industry.

Bibliography

Angelopoulos, A. N. and Bates, S. (2021). A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*.

Mauricio Sadinle, J. L. and Wasserman, L. (2019). Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114(525):223–234.

Mo, Y., Wu, Y., Yang, X., Liu, F., and Liao, Y. (2022). Review the state-of-the-art technologies of semantic segmentation based on deep learning. *Neurocomputing*, 493:626–646.

Papadopoulos, H., Proedrou, K., Vovk, V., and Gammerman, A. (2002). Inductive confidence machines for regression. In *Machine Learning: ECML 2002: 13th European Conference on Machine Learning Helsinki, Finland, August 19–23, 2002 Proceedings 13*, pages 345–356. Springer.

Vovk, V., Gammerman, A., and Shafer, G. (2005). *Algorithmic learning in a random world*, volume 29. Springer.

Title: Methodological Guideline for Conformal Prediction with PUNCC

Keywords: Conformal Prediction, Uncertainty Quantification, Computer Vision, NLP, Anomaly Detection

This methodological guideline presents practical adaptations and applications of Conformal Prediction (CP) using puncc library, which has been integrated in confiance.ai. The aim of this document is to aid the reader to understand the possible usage of CP and puncc through diverse tasks (anomaly detection, classification and segmentation) and scenarios that require robust uncertainty quantification/calibration and provides comprehensive context along with potential results to be expected upon correct use of the proposed approaches.

Our partners: