Conformal Prediction: A Gentle Introduction

Other titles in Foundations and Trends® in Machine Learning

Introduction to Riemannian Geometry and Geometric Statistics: From Basic Theory to Implementation with Geomstats
Nicolas Guigui, Nina Miolane and Xavier Pennec
ISBN: 978-1-63828-154-2

Graph Neural Networks for Natural Language Processing: A Survey
Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei and Bo Long
ISBN: 978-1-63828-142-9

Model-based Reinforcement Learning: A Survey
Thomas M. Moerland, Joost Broekens, Aske Plaat and Catholijn M. Jonker
ISBN: 978-1-63828-056-9

Divided Differences, Falling Factorials, and Discrete Splines: Another Look at Trend Filtering and Related Problems
Ryan J. Tibshirani
ISBN: 978-1-63828-036-1

Risk-Sensitive Reinforcement Learning via Policy Gradient Search
Prashanth L. A. and Michael C. Fu
ISBN: 978-1-63828-026-2

A Unifying Tutorial on Approximate Message Passing
Oliver Y. Feng, Ramji Venkataramanan, Cynthia Rush and Richard J. Samworth
ISBN: 978-1-63828-004-0

Conformal Prediction: A Gentle Introduction

Anastasios N. Angelopoulos
University of California, Berkeley
angelopoulos@berkeley.edu

Stephen Bates
University of California, Berkeley
stephenbates@berkeley.edu

now
the essence of knowledge
Boston — Delft

Foundations and Trends® in Machine Learning

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

A. N. Angelopoulos and S. Bates. *Conformal Prediction: A Gentle Introduction*. Foundations and Trends® in Machine Learning, vol. 16, no. 4, pp. 494–591, 2023.

ISBN: 978-1-63828-159-7

Foundations and Trends® in Machine Learning
Volume 16, Issue 4, 2023

Editorial Board

Editorial Scope

Topics

Foundations and Trends® in Machine Learning publishes survey and tutorial articles in the following topics:

- Adaptive control and signal processing
- Applications and case studies
- Behavioral, cognitive and neural learning
- Bayesian learning
- Classification and prediction
- Clustering
- Data mining
- Dimensionality reduction
- Evaluation
- Game theoretic learning
- Graphical models
- Independent component analysis
- Inductive logic programming
- Kernel methods
- Markov chain Monte Carlo
- Model choice
- Nonparametric methods
- Online learning
- Optimization
- Reinforcement learning
- Relational learning
- Robustness
- Spectral methods
- Statistical learning theory
- Variational inference
- Visualization

Information for Librarians

Foundations and Trends® in Machine Learning, 2023, Volume 16, 6 issues. ISSN paper version 1935-8237. ISSN online version 1935-8245. Also available as a combined paper and online subscription.

Contents

Conformal Prediction: A Gentle Introduction

Anastasios N. Angelopoulos[1] and Stephen Bates[2]

[1] *University of California, Berkeley, USA; angelopoulos@berkeley.edu*
[2] *University of California, Berkeley, USA; stephenbates@berkeley.edu*

ABSTRACT

Black-box machine learning models are now routinely used in high-risk settings, like medical diagnostics, which demand uncertainty quantification to avoid consequential model failures. Conformal prediction (a.k.a. conformal inference) is a user-friendly paradigm for creating statistically rigorous uncertainty sets/intervals for the predictions of such models. Critically, the sets are valid in a *distribution-free* sense: they possess explicit, non-asymptotic guarantees even without distributional assumptions or model assumptions. One can use conformal prediction with any pre-trained model, such as a neural network, to produce sets that are guaranteed to contain the ground truth with a user-specified probability, such as 90%. It is easy-to-understand, easy-to-use, and general, applying naturally to problems arising in the fields of computer vision, natural language processing, deep reinforcement learning, and so on.

This hands-on introduction is aimed to provide the reader a working understanding of conformal prediction and related distribution-free uncertainty quantification techniques with one self-contained document. We lead the reader through practical theory for and examples of conformal prediction

Anastasios N. Angelopoulos and Stephen Bates (2023), "Conformal Prediction: A Gentle Introduction", Foundations and Trends® in Machine Learning: Vol. 16, No. 4, pp 494–591. DOI: 10.1561/2200000101.

and describe its extensions to complex machine learning tasks involving structured outputs, distribution shift, time-series, outliers, models that abstain, and more. Throughout, there are many explanatory illustrations, examples, and code samples in Python. With each code sample comes a Jupyter notebook implementing the method on a real-data example; the notebooks can be accessed and easily run by following the code footnotes.[</>]

[</>] https://github.com/aangelopoulos/conformal-prediction

1

Conformal Prediction

Conformal prediction [72], [88], [116], also known as conformal inference, is a straightforward way to generate prediction sets for any model. We will introduce it with a short, pragmatic image classification example, and follow up in later paragraphs with a general explanation.

The high-level outline of conformal prediction is as follows. First, we begin with a fitted predicted model (such as a neural network classifier) which we will call $\hat{f}$. Then, we create prediction sets (a set of possible labels) for this classifier using a small amount of additional *calibration data*—we will sometimes call this the *calibration step*.

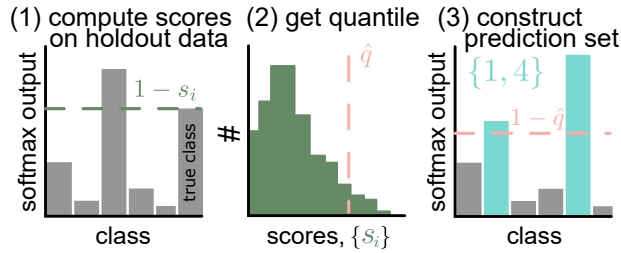
Formally, suppose we have images as input and they each contain one of K classes. We begin with a classifier that outputs estimated probabilities (softmax scores) for each class: $\hat{f}(x) \in [0,1]^K$. Then, we reserve a moderate number (e.g., 500) of fresh i.i.d. pairs of images and classes unseen during training, $(X_1, Y_1), \ldots, (X_n, Y_n)$, for use as calibration data. Using $\hat{f}$ and the calibration data, we seek to construct a *prediction set* of possible labels $\mathcal{C}(X_{\text{test}}) \subset \{1, \ldots, K\}$ that is valid in the following sense:

$$1 - \alpha \leq \mathbb{P}(Y_{\text{test}} \in \mathcal{C}(X_{\text{test}})) \leq 1 - \alpha + \frac{1}{n+1}, \tag{1.1}$$

where $(X_{\text{test}}, Y_{\text{test}})$ is a fresh test point from the same distribution, and $\alpha \in [0, 1]$ is a user-chosen error rate. In words, the probability that the prediction set contains the correct label is almost exactly $1 - \alpha$; we call this property *marginal coverage*, since the probability is marginal (averaged) over the randomness in the calibration and test points. See Figure 1.1 for examples of prediction sets on the Imagenet dataset.

Figure 1.1: Prediction set examples on Imagenet. We show three progressively more difficult examples of the class `fox squirrel` and the prediction sets (i.e., $\mathcal{C}(X_{\text{test}})$) generated by conformal prediction.

To construct $\mathcal{C}$ from $\hat{f}$ and the calibration data, we will perform a simple calibration step that requires only a few lines of code; see the bottom panel of Figure 1.2. We now describe the calibration step in more detail, introducing some terms that will be helpful later on. First, we set the *conformal score* $s_i = 1 - \hat{f}(X_i)_{Y_i}$ to be one minus the softmax output of the true class. The score is high when the softmax output of the true class is low, i.e., when the model is badly wrong. Next comes the critical step: define $\hat{q}$ to be the $\lceil (n+1)(1-\alpha) \rceil / n$ empirical quantile of $s_1, ..., s_n$, where $\lceil \cdot \rceil$ is the ceiling function ($\hat{q}$ is essentially the $1 - \alpha$ quantile, but with a small correction). Finally, for a new test data point (where X_{test} is known but Y_{test} is not), create a prediction set $\mathcal{C}(X_{\text{test}}) = \{y : \hat{f}(X_{\text{test}})_y \geq 1 - \hat{q}\}$ that includes all classes with a high enough softmax output (see Figure 1.2). Remarkably, this algorithm gives prediction sets that are guaranteed to satisfy (1.1), no matter what (possibly incorrect) model is used or what the (unknown) distribution of the data is.




```python
# 1: get conformal scores. n = calib_Y.shape[0]
cal_smx = model(calib_X).softmax(dim=1).numpy()
cal_scores = 1-cal_smx[np.arange(n),cal_labels]
# 2: get adjusted quantile
q_level = np.ceil((n+1)*(1-alpha))/n
qhat = np.quantile(cal_scores, q_level, method='higher')
val_smx = model(val_X).softmax(dim=1).numpy()
prediction_sets = val_smx >= (1-qhat) # 3: form prediction sets
```

Figure 1.2: Illustration of conformal prediction with Python code.[</>]

Remarks

Let us think about the interpretation of $\mathcal{C}$. The function $\mathcal{C}$ is *set-valued*—it takes in an image, and it outputs a set of classes as in Figure 1.1. The model's softmax outputs help to generate the set. This method constructs a different output set *adaptively to each particular input.* The sets become larger when the model is uncertain or the image is intrinsically hard. This is a property we want, because the size of the set gives you an indicator of the model's certainty. Furthermore, $\mathcal{C}(X_{\text{test}})$ can be interpreted as a set of plausible classes that the image X_{test} could be assigned to. Finally, $\mathcal{C}$ is *valid*, meaning it satisfies (1.1).[1] These properties of $\mathcal{C}$ translate naturally to other machine learning problems, like regression, as we will see.

[</>] https://github.com/aangelopoulos/conformal-prediction/blob/main/notebooks/imagenet-smallest-sets.ipynb

[1] Due to the discreteness of Y, a small modification involving tie-breaking is needed to additionally satisfy the upper bound (see Angelopoulos *et al.* [6] for details; this randomization is usually ignored in practice). We will henceforth ignore such tie-breaking.

With an eye towards generalization, let us review in detail what happened in our classification problem. To begin, we were handed a model that had an inbuilt, but heuristic, notion of uncertainty: softmax outputs. The softmax outputs attempted to measure the conditional probability of each class; in other words, the jth entry of the softmax vector estimated $\mathbb{P}(Y = j \mid X = x)$, the probability of class j conditionally on an input image x. However, we had no guarantee that the softmax outputs were any good; they may have been arbitrarily overfit or otherwise untrustworthy. Therefore, instead of taking the softmax outputs at face value, we used the holdout set to adjust for their deficiencies.

The holdout set contained $n \approx 500$ fresh data points that the model never saw during training, which allowed us to get an honest appraisal of its performance. The adjustment involved computing conformal scores, which grow when the model is uncertain, but are not valid prediction intervals on their own. In our case, the conformal score was one minus the softmax output of the true class, but in general, the score can be any function of x and y. We then took $\hat{q}$ to be roughly the $1 - \alpha$ quantile of the scores. In this case, the quantile had a simple interpretation—when setting $\alpha = 0.1$, at least 90% of ground truth softmax outputs are guaranteed to be above the level $1 - \hat{q}$ (we prove this rigorously in Appendix D). Taking advantage of this fact, at test-time, we got the softmax outputs of a new image X_{test} and collected all classes with outputs above $1 - \hat{q}$ into a prediction set $\mathcal{C}(X_{\text{test}})$. Since the softmax output of the new true class Y_{test} is guaranteed to be above $1 - \hat{q}$ with probability at least 90%, we finally got the guarantee in Eq. (1.1).

1.1 Instructions for Conformal Prediction

As we said during the summary, conformal prediction is not specific to softmax outputs or classification problems. In fact, conformal prediction can be seen as a method for taking **any heuristic notion of uncertainty** from **any model** and converting it to a rigorous one (see the diagram in Figure 1.3). Conformal prediction does not care if the underlying prediction problem is discrete/continuous or classification/regression.

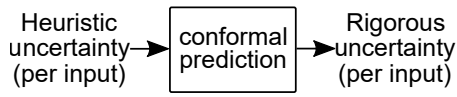


Figure 1.3: Conformal prediction converts heuristic notions of uncertainty into rigorous ones.

We next outline conformal prediction for a general input x and output y (not necessarily discrete).

1. Identify a heuristic notion of uncertainty using the pre-trained model.
2. Define the score function $s(x, y) \in \mathbb{R}$. (Larger scores encode worse agreement between x and y).
3. Compute $\hat{q}$ as the $\frac{\lceil (n+1)(1-\alpha) \rceil}{n}$ quantile of the calibration scores $s_1 = s(X_1, Y_1), ..., s_n = s(X_n, Y_n)$.
4. Use this quantile to form the prediction sets for new examples:

$$\mathcal{C}(X_{\text{test}}) = \{y : s(X_{\text{test}}, y) \leq \hat{q}\} . \tag{1.2}$$

As before, these sets satisfy the validity property in (1.1), for any (possibly uninformative) score function and (possibly unknown) distribution of the data. We formally state the coverage guarantee next.

Theorem 1.1 (Conformal coverage guarantee; Vovk, Gammerman, and Saunders [117])**.** Suppose $(X_i, Y_i)_{i=1,...,n}$ and $(X_{\text{test}}, Y_{\text{test}})$ are i.i.d. and define $\hat{q}$ as in step 3 above and $\mathcal{C}(X_{\text{test}})$ as in step 4 above. Then the following holds:

$$P\Big(Y_{\text{test}} \in \mathcal{C}(X_{\text{test}})\Big) \geq 1 - \alpha.$$

See Appendix D for a proof and a statement that includes the upper bound in (1.1). We note that the above is only a special case of conformal prediction, called *split conformal prediction*. This is the most widely-used version of conformal prediction, and it will be our primary focus. To complete the picture, we describe conformal prediction in full generality later in Section 6 and give an overview of the literature in Section 7.

Choice of Score Function

Upon first glance, this seems too good to be true, and a skeptical reader might ask the following question:

How is it possible to construct a statistically valid prediction set even if the heuristic notion of uncertainty of the underlying model is arbitrarily bad?

Let's give some intuition to supplement the mathematical understanding from the proof in Appendix D. Roughly, if the scores s_i correctly rank the inputs from lowest to highest magnitude of model error, then the resulting sets will be smaller for easy inputs and bigger for hard ones. If the scores are bad, in the sense that they do not approximate this ranking, then the sets will be useless. For example, if the scores are random noise, then the sets will contain a random sample of the label space, where that random sample is large enough to provide valid marginal coverage. This illustrates an important underlying fact about conformal prediction: although the guarantee always holds, **the usefulness of the prediction sets is primarily determined by the score function**. This should be no surprise—the score function incorporates almost all the information we know about our problem and data, including the underlying model itself. For example, the main difference between applying conformal prediction on classification problems versus regression problems is the choice of score. There are also many possible score functions for a single underlying model, which have different properties. Therefore, constructing the right score function is an important engineering choice. We will next show a few examples of good score functions.

Acknowledgements

The authors are grateful to the editors and reviewers of Foundations and Trends in Machine Learning for their extensive feedback. A. N. A. was partially supported by the National Science Foundation Graduate Research Fellowship Program and a Berkeley Fellowship. The authors would like to thank numerous colleagues for their correspondence which led to improvements in our work; the non-exhaustive list includes Rina Barber, Emmanuel Candès, Todd Chapman, John Cherian, Giovanni Cherubin, Nandita Damaraju, Tiffany Ding, Edgar Dobriban, Clara Fannjiang, Adam Fisch, Alexander Gammerman, Isaac Gibbs, Patrizio Giovannotti, Leying Guan, Michael I. Jordan, Roger Koenker, Amit Kohli, Lihua Lei, Valeriy Manokhin, Andrea Panizza, Ilija Radosavović, Aaditya Ramdas, Yaniv Romano, Aaron Roth, Tal Schuster, Ryan Tibshirani, Vladimir Vovk, Mariel Werner, Christopher T. Yeh, and Tijana Zrnić.

Appendices

A

Distribution-Free Control of General Risks

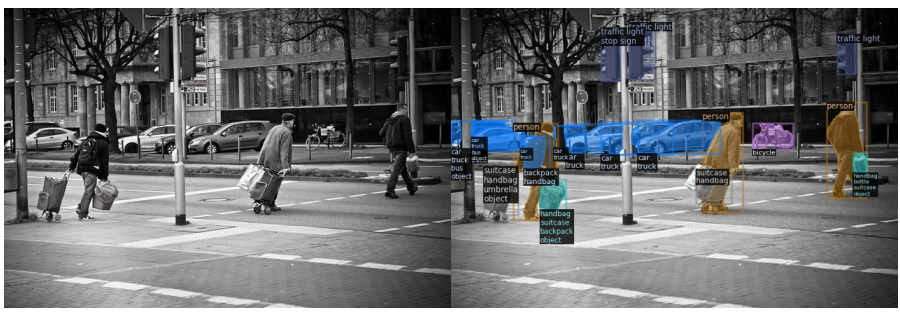


Figure A.1: Object detection with simultaneous distribution-free guarantees on the expected intersection-over-union, recall, and coverage rate.

For many prediction tasks, the relevant notion of reliability is not coverage. Indeed, many applications have problem-specific performance metrics—from false-discovery rate to fairness—that directly encode the soundness of a prediction. In Section 4.3, we saw how to control the expectation of monotone loss functions using conformal risk control. Here, we generalize further to control *any* risk and multiple risks in a distribution-free way without retraining the model. As an example, in instance segmentation, we are given an image and asked to identify all

objects in the image, segment them, and classify them. All three of these sub-tasks have their own risks: recall, *intersection-over-union* (IOU), and coverage respectively. These risks can be automatically controlled using distribution-free statistics, as we preview in Figure A.1.

We first re-introduce the theory of risk control below, then give a list of illustrative examples. As in conformal risk control, we start with a pretrained model $\hat{f}$. The model also has a *parameter* λ, which we are free to choose. We use $\hat{f}(x)$ and λ to form our prediction, $\mathcal{T}_\lambda(x)$, which may be a set or some other object. For example, when performing regression, λ could threshold the estimated probability density, as in Figure A.2.

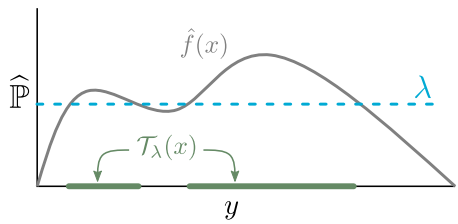


Figure A.2: A family of prediction sets produced by thresholding an estimated probability density.

We then define a notion of risk $R(\lambda)$. The risk function measures the quality of $\mathcal{T}_\lambda$ according to the user. The goal of risk control is to use our calibration set to pick a parameter $\hat{\lambda}$ so that the risk is small with high probability. In formal terms, for a user-defined *risk tolerance* α and *error rate* δ, we seek to ensure

$$\mathbb{P}\left(R(\hat{\lambda}) < \alpha\right) \geq 1 - \delta, \tag{A.1}$$

where the probability is taken over the calibration data used to pick $\hat{\lambda}$. Note that this guarantee is high-probability, unlike that in Section 4.3, which is in expectation. We will soon introduce a distribution-free technique called *Learn then Test* (LTT) for finding $\hat{\lambda}$ that satisfy (A.1). Below we include two example applications of risk control which would be impossible with conformal prediction and conformal risk control.

- *Multi-label Classification with FDR Control:* In this setting, X_{test} is an image and Y_{test} is a subset of K classes contained in the

image. Our model $\hat{f}$ gives us the probability each of the K classes is contained in the image. We will include a class in our estimate of y if $\hat{f}_k > \lambda$ — i.e., the parameter λ thresholds the estimated probabilities. We seek to find the $\hat{\lambda}$s that guarantees our predicted set of labels is sufficiently reliable as measured by the *false-discovery rate* (FDR) risk $R(\hat{\lambda})$.

- *Simultaneous Guarantees on OOD Detection and Coverage:* For each input X_{test} with true class Y_{test}, we want to decide if it is out-of-distribution. If so, we will flag it as such. Otherwise, we want to output a prediction set that contains the true class with 90% probability. In this case, we have two models: $\text{OOD}(x)$, which tells us how OOD the input is, and $\hat{f}(x)$, which gives the estimated probability that the input comes from each of K classes. In this case, λ has two coordinates, and we also have two risks. The first coordinate λ_1 tells us where to threshold $\text{OOD}(x)$ such that the fraction of false alarms R_1 is controlled. The second coordinate λ_2 tells us how many classes to include in the prediction set to control the miscoverage R_2 among points identified as in-distribution. We will find $\hat{\lambda}$s that control both $R_1(\hat{\lambda})$ and $R_2(\hat{\lambda})$ jointly.

We will describe each of these examples in detail in Appendix B. Many more worked examples, including the object detection example in Figure A.1, are available in the cited literature on risk control [2], [10]. First, however, we will introduce the general method of risk control via Learn then Test.

A.1 Instructions for Learn then Test

First, we will describe the formal setting of risk control. We introduce notation and the risk-control property in Definition A.1. Then, we describe the calibration algorithm.

Formal Notation for Error Control

Let $(X_i, Y_i)_{i=1,\ldots,n}$ be an independent and identically distributed (i.i.d.) set of variables, where the features X_i take values in $\mathcal{X}$ and the responses

Y_i take values in $\mathcal{Y}$. The researcher starts with a pre-trained predictive model $\hat{f}$. We show how to subsequently create predictors from $\hat{f}$ that control a risk, regardless of the quality of the initial model fit or the distribution of the data.

Next, let $\mathcal{T}_\lambda : \mathcal{X} \to \mathcal{Y}'$ be a function with parameter λ that maps a feature to a prediction ($\mathcal{Y}'$ can be any space, including the space of responses $\mathcal{Y}$ or prediction sets $2^{\mathcal{Y}}$). This function $\mathcal{T}_\lambda$ would typically be constructed from the predictive model, $\hat{f}$, as in our earlier regression example. We further assume λ takes values in a (possibly multidimensional) discrete set Λ. If Λ is not naturally discrete, we usually discretize it finely. For example, Λ could be the set $\{0, 0.001, 0.002, ..., 0.999, 1\}$.

We then allow the user to choose a *risk* for the predictor $\mathcal{T}_\lambda$. This risk can be any function of $\mathcal{T}_\lambda$, but often we take the risk function to be the expected value of a *loss function*,

$$R(\mathcal{T}_\lambda) = \mathbb{E}\left[\underbrace{L(\mathcal{T}_\lambda(X_{\text{test}}), Y_{\text{test}})}_{\text{Loss function}}\right]. \tag{A.2}$$

The loss function is a deterministic function that is high when $\mathcal{T}_\lambda(X_{\text{test}})$ does badly at predicting Y_{test}. The risk then averages this loss over the distribution of $(X_{\text{test}}, Y_{\text{test}})$. For example, taking

$$R_{\text{miscoverage}}(\mathcal{T}_\lambda) = \mathbb{E}\left[\mathbb{1}\left\{Y_{\text{test}} \notin \mathcal{T}_\lambda(X_{\text{test}})\right\}\right] = \mathbb{P}\left(Y_{\text{test}} \notin \mathcal{T}_\lambda(X_{\text{test}})\right)$$

gives us the familiar case of controlling miscoverage.

To aid the reader, we point out some facts about (A.2) that may not be obvious. The input $\mathcal{T}_\lambda$ to the risk is a function; this makes the risk a *functional* (a function of a function). When we plug $\mathcal{T}_\lambda$ into the risk, we take an expectation of the loss over the randomness in a single test point. At the end of the process, for a deterministic λ, we get a deterministic scalar $R(\mathcal{T}_\lambda)$. Henceforth, for ease of notation, we abbreviate this number as $R(\lambda) := R(\mathcal{T}_\lambda)$.

Our goal is control the risk in the following sense:

Definition A.1 (Risk control)**.** Let $\hat{\lambda}$ be a random variable taking values in Λ (i.e., the output of an algorithm run on the calibration data). We say that $\mathcal{T}_{\hat{\lambda}}$ is a (α, δ)-*risk-controlling prediction* (RCP) if, with probability at least $1 - \delta$, we have $R(\hat{\lambda}) \leq \alpha$; see Figure A.3.

In Definition A.1, we plug in a *random parameter* $\hat{\lambda}$ which is chosen based on our calibration data; therefore, $R(\hat{\lambda})$ is random even though the risk is a deterministic function. The high-probability portion of Definition A.1 therefore says that $\hat{\lambda}$ can only violate risk control if we receive a bad calibration set; this happens with probability at most δ. The distribution of the risk over many resamplings of the calibration data should therefore look as below.

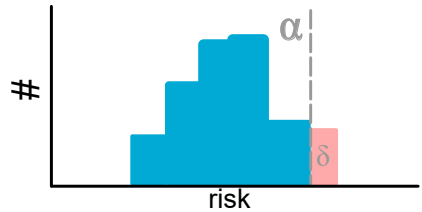


Figure A.3: The risk distribution of an RCP.

The Learn then Test Procedure

Recalling Definition A.1, our goal is to find a set function whose risk is less than some user-specified threshold α. To do this, we search across the collection of functions $\{\mathcal{T}_\lambda\}_{\lambda\in\Lambda}$ and estimate their risk on the calibration data $(X_1, Y_1), \ldots, (X_n, Y_n)$. The output of the procedure will be a set of λ values which are all guaranteed to control the risk, $\widehat{\Lambda} \subseteq \Lambda$. The Learn then Test procedure is outlined below.

1. For each $\lambda \in \Lambda$, associate the null hypothesis $\mathcal{H}_\lambda : R(\lambda) > \alpha$. Notice that *rejecting* the $\mathcal{H}_\lambda$ means you selected λ as a point where the risk is controlled. In Figure A.4, we denote each null with a blue dot; the yellow dot is highlighted, so we can keep track of it as we explain the procedure.

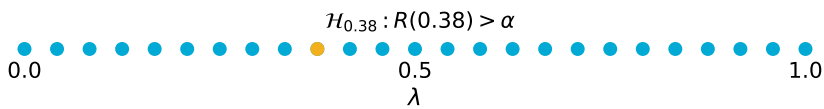


Figure A.4: A grid of null hypotheses.

2. For each null hypothesis, compute a p-value using a concentration inequality. For example, Hoeffding's inequality yields $p_\lambda = e^{-2n(\alpha - \widehat{R}(\lambda))_+^2}$, where $\widehat{R}(\lambda) = \frac{1}{n}\sum_{i=1}^{n} L(\mathcal{T}_\lambda(X_i), Y_i)$; see Figure A.5. We remind the reader what a p-value is, why it is relevant to risk control, and point to references with stronger p-values in A.1.1.

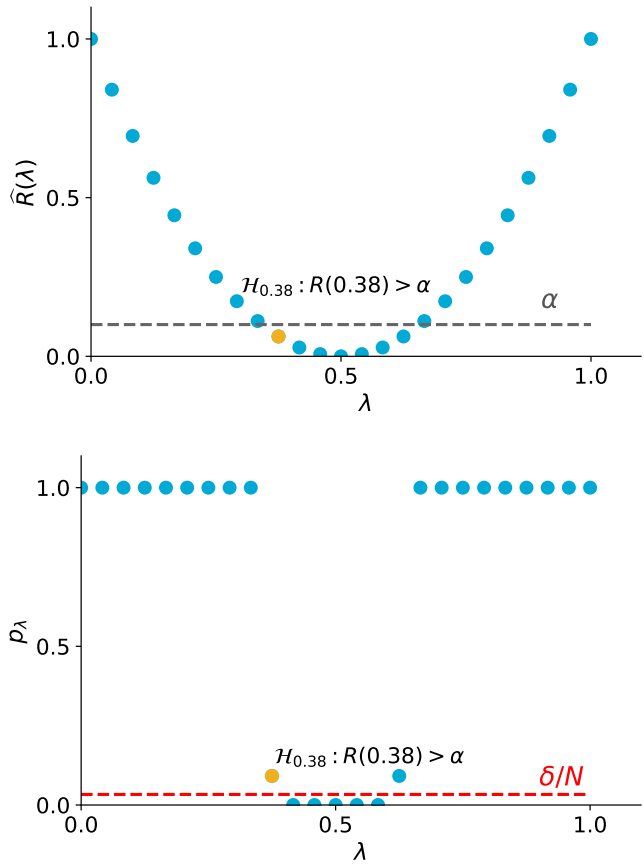


Figure A.5: Calculate a p-value for each null.

3. Return $\widehat{\Lambda} = \mathcal{A}(\{p_\lambda\}_{\lambda \in \Lambda})$, where $\mathcal{A}$ is an algorithm that controls the familywise-error rate (FWER). For example, the Bonferroni

correction yields $\widehat{\Lambda} = \{\lambda : p_\lambda < \frac{\delta}{|\Lambda|}\}$; see Figure A.6. We define the FWER and preview ways to design good FWER-controlling procedures in Appendix A.1.2. In Figure A.6, nulls with red crosses through them below have been rejected by the procedure; i.e., they all control the risk with high probability.

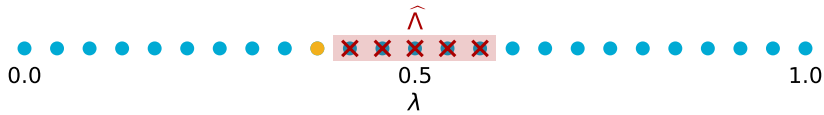


Figure A.6: Reject the p-values that pass the FWER-control algorithm. These values of λ simultaneously control the risk.

By following the above procedure, we get the statistical guarantee in Theorem A.1.

Theorem A.1. The $\widehat{\Lambda}$ returned by the Learn then Test procedure satisfies

$$\mathbb{P}\left(\sup_{\hat{\lambda}\in\widehat{\Lambda}}\{R(\hat{\lambda})\} \leq \alpha\right) \geq 1 - \delta.$$

Thus, selecting any $\hat{\lambda} \in \widehat{\Lambda}$, $\mathcal{T}_{\hat{\lambda}}$ is an (α, δ)-RCP. See Figure A.7 for an algorithm.

The LTT procedure decomposes risk control into two subproblems: computing p-values and combining them with multiple testing. We will now take a closer look at each of these subproblems.

A.1.1 Crash Course on Generating p-values

What is a p-value, and why is it related to risk control? In Step 1 of the LTT procedure, we associated a null hypothesis $\mathcal{H}_\lambda$ to every $\lambda \in \Lambda$. When the null hypothesis at λ holds, the risk is *not* controlled for that value of the parameter. In this reframing, our task is to automatically identify points λ where the null hypothesis does not hold—i.e., to *reject the null hypotheses* for some subset of λ such that

```
#Implementation of LTT.
# Assume access to X, Y where n=X.shape[0]=Y.shape[0]
lambdas = torch.linspace(0,1,N) # Commonly choose N=1000
losses = torch.zeros((n,N)) # Compute the loss function next
for (i,j) in [(i,j) for i in range(n) for j in range(N)]:
  prediction_set = T(X[i],lambdas[j]) # T ( ) is problem-dependent
  losses[i,j] = get_loss(prediction_set,Y[i]) # Problem-dependent
risk = losses.mean(dim=0)
pvals = torch.exp(-2*n*(torch.relu(alpha-risk)**2)) # Or any p-value
lambda_hat = lambdas[pvals<delta/lambdas.shape[0]]
                    # Or any FWER-controlling algorithm
```

Figure A.7: PyTorch code for running Learn then Test.

$R(\lambda) \leq \alpha$. The process of accepting or rejecting a null hypothesis is called *hypothesis testing*.

Rejecting the null hypothesis $\mathcal{H}_\lambda \rightarrow$ the risk *is* controlled at λ.
Accepting the null hypothesis $\mathcal{H}_\lambda \rightarrow$ the risk *is not* controlled at λ.

In order to reject a null hypothesis, we need to have empirical evidence that at λ, the risk is controlled. We use our calibration data to summarize this information in the form of a *p-value* p_λ. A p-value must satisfy the following condition, which we sometimes refer to as *validity* or *super-uniformity*,

$$\forall t \in [0,1], \;\; \mathbb{P}_{\mathcal{H}_\lambda}(p_\lambda \leq t) \leq t,$$

where $\mathbb{P}_{\mathcal{H}_\lambda}$ refers to the probability under the null hypothesis. Parsing the super-uniformity condition carefully tells us that when p_λ is low, there is evidence against the null hypothesis $\mathcal{H}_\lambda$. In other words, for a particular λ, we can reject $\mathcal{H}_\lambda$ if $p_\lambda < 5\%$ and expect to be wrong no more than 5% of the time; see Figure A.8 for a graphical representation. This process is called *testing the hypothesis at level* δ, where in the previous sentence, $\delta = 5\%$.

One of the key ingredients in Learn then Test is a p-value with distribution-free validity: it is valid under without assumptions on the data distribution. For example, when working with risk functions that take values in $[0, 1]$—like coverage, IOU, FDR, and so on—the easiest choice of p-value is based on Hoeffding's inequality:

$$p_\lambda^{\text{Hoeffding}} = e^{-2n\left(\alpha - \widehat{R}(\lambda)\right)_+^2}.$$

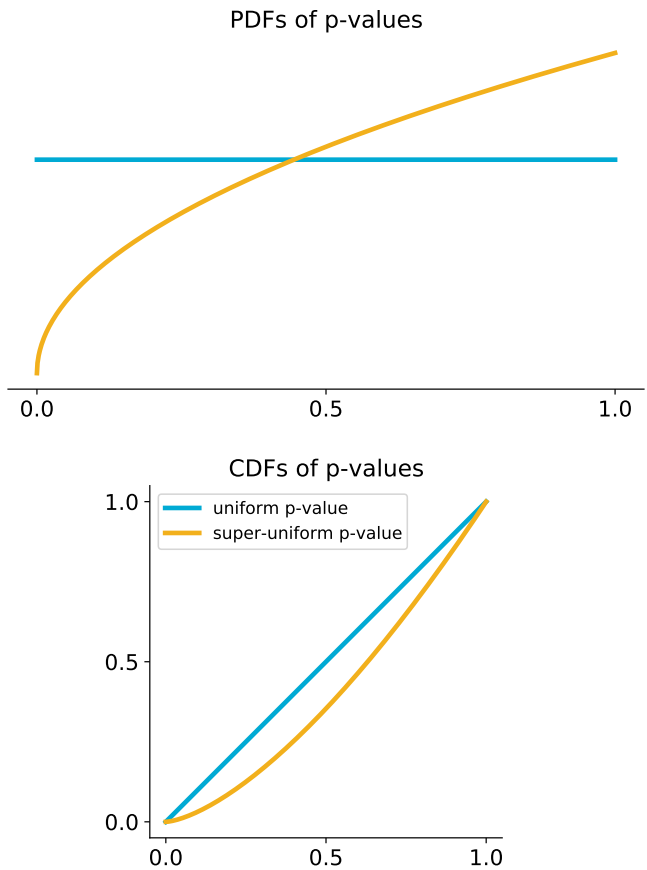


Figure A.8: PDF and CDF of valid p-values.

More powerful p-values based on tighter concentration bounds are included in Angelopoulos *et al.* [2]. In particular, many of the practical

examples in that reference use a stronger p-value called the *Hoeffding-Bentkus* (HB) p-value,

$$p_\lambda^{\mathrm{HB}} = \min\left(\exp\{-nh_1(\widehat{R}(\lambda) \wedge \alpha, \alpha)\}, e\mathbb{P}(\mathrm{Bin}(n,\alpha) \leq \left\lceil n\widehat{R}(\lambda)\right\rceil)\right),$$
$$\text{where } h_1(a,b) = a\log\left(\frac{a}{b}\right) + (1-a)\log\left(\frac{1-a}{1-b}\right).$$

Note that any valid p-value will work—it is fine for the reader to keep $p_\lambda^{\mathrm{Hoeffding}}$ in mind for the rest of this work, with the understanding that more powerful choices are available.

A.1.2 Crash Course on Familywise-Error Rate Algorithms

If we only had one hypothesis H_λ, we could simply test it at level δ. However, we have one hypothesis for each $\lambda \in \Lambda$, where $|\Lambda|$ is often very large (in the millions or more). This causes a problem: the more hypotheses we test, the higher chance we incorrectly reject at least one hypothesis. We can formally reason about this with the *familywise-error rate* (FWER).

Definition A.2 (familywise-error rate)**.** The familywise-error rate of a procedure returning $\hat{\Lambda}$ is the probability of making at least one false rejection, i.e.,

$$\mathrm{FWER}\left(\widehat{\Lambda}\right) = \mathbb{P}\left(\exists\hat{\lambda} \in \widehat{\Lambda} : R(\hat{\lambda}) > \alpha\right).$$

As a simple example to show how naively thresholding the p-values at level δ fails to control FWER, consider the case where all the hypotheses are null, and we have uniform p-values independently tested at level δ. The FWER then approaches 1; see below.

$$\text{If we take } \widehat{\Lambda} = \{\lambda : p_\lambda < \delta\}, \text{ then } \mathrm{FWER}(\widehat{\Lambda}) = 1 - (1-\delta)^{|\Lambda|}.$$

This simple toy analysis exposes a deeper problem: without an intelligent strategy for combining the information from many p-values together, we can end up making false rejections with high probability. Our challenge is to intelligently combine the p-values to avoid this issue of multiplicity (without assuming the p-values are independent).

This fundamental statistical challenge has led to a decades-long and continually rich area of research called *multiple hypothesis testing*. In

particular, a genre of algorithms called *FWER-controlling algorithms* seek to select the largest set of $\widehat{\Lambda}$ that guarantees $\text{FWER}(\widehat{\Lambda}) \leq \delta$. The simplest FWER-controlling algorithm is the *Bonferroni correction*,

$$\widehat{\Lambda}_{\text{Bonferroni}} = \left\{ \lambda \in \Lambda : p_\lambda \leq \frac{\delta}{|\Lambda|} \right\}.$$

Under the hood, the Bonferroni correction simply tests each hypothesis at level $\delta/|\Lambda|$, so the probability there exists a failed test is no more than δ by a union bound. It should not be surprising that there exist improvements on Bonferroni correction.

First, we will discuss one important improvement in the case of a monotone loss function: *fixed-sequence testing*. As the name suggests, in fixed-sequence testing, we construct a sequence of hypotheses $\{\mathcal{H}_{\lambda_j}\}_{j=1}^{N}$ where $\text{N} = |\Lambda|$, before looking at our calibration data. Usually, we just sort our hypotheses from most- to least-promising based on information we knew a-priori. For example, if large values of λ are more likely to control the risk, $\{\lambda_j\}_{j=1}^{N}$ just sorts Λ from greatest to least. Then, we test the hypotheses sequentially in some fixed order at level δ, including them in $\widehat{\Lambda}$ as we go, and stopping when we make our first acceptance, as we illustrate below in Figure A.9:

$$\widehat{\Lambda}_{\text{FST}} = \{\lambda_j, j \leq T\}, \text{ where } T = \max\left\{t \in \{1, ..., N\} : p_{\lambda_{t'}} \leq \delta, \text{ for all } t' \leq t\right\}$$

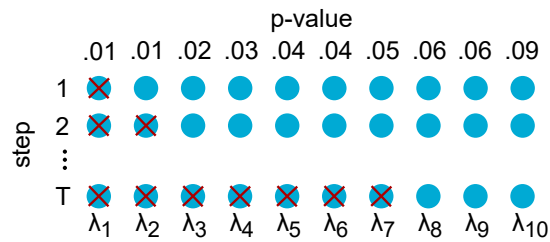


Figure A.9: An example of fixed-sequence testing with $\delta = 0.05$. Each blue circle represents a null, and each row a step of the procedure. The nulls with a red cross have been rejected at that step.

This sequential procedure, despite testing all hypotheses it encounters at level δ, still controls the FWER. For monotone and near-monotone risks, such as the false-discovery rate, it works quite well.

It is also possible to extend the basic idea of fixed-sequence testing to non-monotone functions, creating powerful and flexible FWER-controlling procedures using an idea called sequential graphical testing [13]. Good graphical FWER-controlling procedures can be designed to have high power for particular problems, or alternatively, automatically discovered using data. This topic is given a detailed treatment in Angelopoulos *et al.* [2], and we omit it here for simplicity.

We have described a general-purpose pipeline for distribution-free risk control. It is described in PyTorch code in Figure A.7. Once the user sets up the problem (i.e., picks Λ, $\mathcal{T}_\lambda$, and R), the LTT pipeline we described above automatically produces $\widehat{\Lambda}$. We now go through three worked examples which teach the reader how to choose Λ, $\mathcal{T}$ and R in practical circumstances.

B

Examples of Distribution-Free Risk Control

In this section, we will walk through several examples of distribution-free risk control applied to practical machine learning problems. The goal is again to arm the reader with an arsenal of pragmatic prototypes of distribution-free risk control that work on real problems.

B.1 Multi-label Classification with FDR Control

We begin our sequence of examples with a familiar and fundamental setup: multi-label classification. Here, the features X_{test} can be anything (e.g. an image), and the label $Y_{\text{test}} \subseteq \{1, ..., K\}$ must be a set of classes (e.g. those contained in the image X_{test}). We have a pre-trained machine learning model $\hat{f}(x)$, which gives us an estimated probability $\hat{f}(x)_k$ that class k is in the corresponding set-valued label. We will use these probabilities to include the estimated most likely classes in our prediction set,

$$\mathcal{T}_\lambda(x) = \{k : \hat{f}(x)_k > \lambda\}, \quad \lambda \in \Lambda$$

where $\Lambda = \{0, 0.001, ..., 1\}$ (a discretization of $[0, 1]$). However, one question remains: *how do we choose* λ?

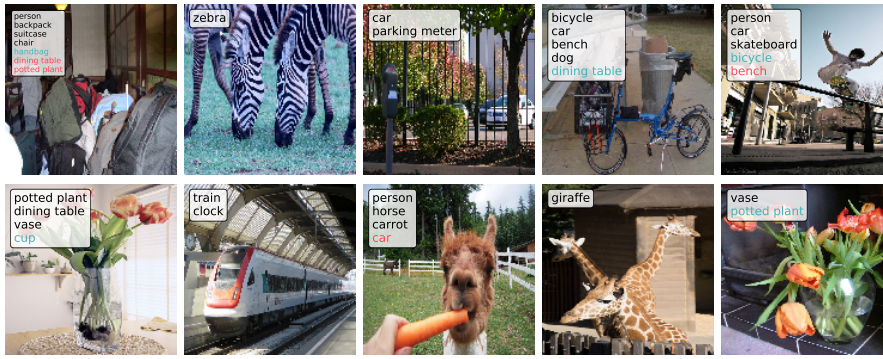


Figure B.1: Examples of multi-label classification with FDR control on the MS-COCO dataset. Black classes are true positives, blue classes are spurious, and red classes are missed. The FDR is controlled at level $\alpha = 0.1$, $\delta = 0.1$.

LTT will allow us to identify values of λ that satisfy a precise probabilistic guarantee—in this case, a bound on the *false-discovery rate* (FDR),

$$R_{\text{FDR}}(\lambda) = \mathbb{E}\left[\underbrace{1 - \frac{|Y_{\text{test}} \cap \mathcal{T}_\lambda(X_{\text{test}})|}{|\mathcal{T}_\lambda(X_{\text{test}})|}}_{L_{\text{FDP}}(\mathcal{T}_\lambda(X_{\text{test}}), Y_{\text{test}})}\right].$$

As annotated in the underbrace, the FDR is the expectation of a loss function, the *false-discovery proportion* (FDP). The FDP is low when our prediction set $\mathcal{T}_\lambda(X_{\text{test}})$ contains mostly elements from Y_{test}. In this sense, the FDR measures the quality of our prediction set: if we have a low FDR, it means most of the elements in our prediction set are good. By setting $\alpha = 0.1$ and $\delta = 0.1$, we desire that

$$\mathbb{P}\left[R_{\text{FDR}}(\hat{\lambda}) > 0.1\right] < 0.1,$$

where the probability is over the randomness in the calibration set used to pick $\hat{\lambda}$.

Now that we have set up our problem, we can just run the LTT procedure via the code in Figure B.2. We use fixed-sequence testing because the FDR is a nearly monotone risk. In practice, we also wish to use the HB p-value, which is stronger than the simple Hoeffding p-value

```
# model is a multi-class neural network, X.shape[0]=Y.shape[0]=n
lambdas = torch.linspace(0,1,N) # N can be taken to inf without penalty
losses = torch.zeros((n,N)) # loss for example i at lambdas[j]
for i in range(n): # In reality we parallelize these loops
  sigmoids = model(X[i].unsqueeze(0)).sigmoid().squeeze()
  for j in range(N):
    T = sigmoids > lambdas[j] # This is the prediction set
    set_size = T.float().sum()
    if set_size != 0:
      losses[i,j] = 1 - (T[Y] == True).float().sum()/set_size
risk = losses.mean(dim=0)
pvals = torch.exp(-2*n*(torch.relu(alpha-risk)**2)) # Or any p-value
# Fixed-sequence test going backwards from lambdas[-1]
below_delta = (pvals <= delta).float()
valid = torch.tensor(
            [(below_delta[j:].mean() == 1) for j in range(N)]
        )
lambda_hat = lambdas[valid]
```

Figure B.2: PyTorch code for performing FDR control with LTT.

in Figure B.2. The result of this procedure on the MS-COCO image dataset is in Figure B.1.

B.2 Simultaneous Guarantees on OOD Detection and Coverage

In our next example, we perform classification with two goals:

1. Flag *out-of-distribution* (OOD) inputs without too many false flags.

2. If an input is deemed *in-distribution* (In-D), output a prediction set that contains the true class with high probability.

Part of the purpose of this example is to teach the reader how to deal with multiple risk functions (one of which is a conditional risk) and a multi-dimensional parameter λ.

Our setup requires two different models. The first, $\text{OOD}(x)$, outputs a scalar that should be larger when the input is OOD. The second, $\hat{f}(x)_y$, estimates the probability that input x is of class y; for example, $\hat{f}(x)$ could represent the softmax outputs of a neural net. Similarly, the construction of $\mathcal{T}_\lambda(x)$ has two substeps, each of which uses a different

model. In our first substep, when $\text{OOD}(x)$ becomes sufficiently large, exceeding λ_1, we flag the example as OOD by outputting $\emptyset$. Otherwise, we essentially use the APS method from Section 2.1 to form prediction sets. We precisely describe this procedure below:

$$\mathcal{T}_\lambda(x) = \begin{cases} \emptyset & \text{OOD}(x) > \lambda_1 \\ \{\pi_1(x), ..., \pi_K(x)\} & \text{else,} \end{cases}$$

where $K = \inf\{k : \sum_{j=1}^{k} \hat{f}(x)_{\pi_j(x)} > \lambda_2\}$ and $\pi(x)$ sorts $\hat{f}(x)$ from greatest to least. We usually take $\Lambda = \{0, 1/N, 2/N, ..., 1\}^2$, i.e., we discretize the box $[0,1] \times [0,1]$ into N^2 smaller boxes, with $N \approx 1000$. The intuition of $\mathcal{T}_\lambda(x)$ is very simple. If the example is sufficiently atypical, we give up. Otherwise, we create a prediction set using a procedure similar to (but not identical to) conformal prediction; see Figure B.3.

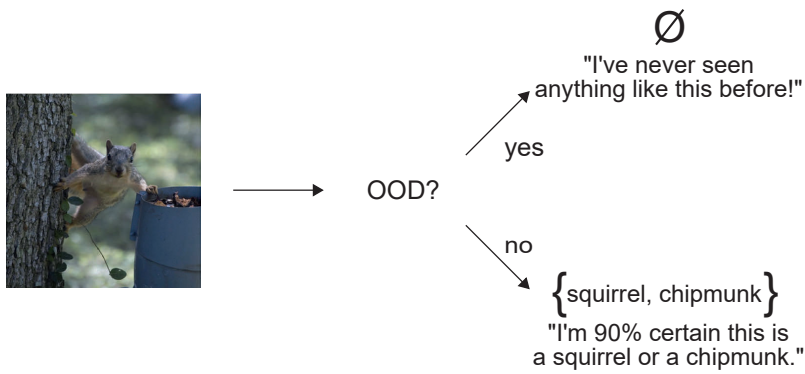


Figure B.3: Out-of-distribution detection with prediction sets.

Along the same lines, we control two risk functions simultaneously,

$$R_1(\lambda) = \mathbb{P}\left(\mathcal{T}_\lambda(X_{\text{test}}) = \emptyset\right) \text{ and } R_2(\lambda) = \mathbb{P}\left(Y_{\text{test}} \notin \mathcal{T}_\lambda(X_{\text{test}}) \,\middle|\, \mathcal{T}_\lambda(X_{\text{test}}) \neq \emptyset\right)$$

The first risk function R_1 is the probability of a false flag, and the second risk function R_2 is the coverage conditionally on being deemed in-distribution. The user must define risk-tolerances for each, so α is a two-vector, where α_1 determines the desired fraction of false flags and

α_2 determines the desired miscoverage rate. Setting $\alpha = (0.05, 0.1)$ will guarantee that we falsely throw out no more than 5% of in-distribution data points, and also that among the data points we claim are in-distribution, we will output a prediction set containing the correct class with 90% probability. In order to control both risks, we now need to associate a composite null hypothesis to each $\lambda \in \Lambda$. Namely, we choose

$$\mathcal{H}_\lambda : \mathcal{H}_\lambda^{(1)} \text{ or } \mathcal{H}_\lambda^{(2)},$$

where $\mathcal{H}_\lambda$ is the union of two intermediate null hypotheses,

$$\mathcal{H}_\lambda^{(1)} : R_1(\lambda) > \alpha_1 \text{ and } \mathcal{H}_\lambda^{(2)} : R_2(\lambda) > \alpha_2.$$

We summarize our setup in Table B.1.

Table B.1

Goal	Null hypothesis	Parameter
Few false positives	$H_\lambda^{(1)} : R_1(\lambda) > \alpha_1$	λ_1
Coverage of prediction sets	$H_\lambda^{(2)} : R_2(\lambda) > \alpha_2$	λ_2

Having completed our setup, we can now apply LTT. The presence of multiple risks creates some wrinkles, which we will now iron out with the reader. The null hypothesis $\mathcal{H}_\lambda$ has a different structure than the ones we saw before, but we can use the same tools to test it. To start, we produce p-values for the intermediate nulls,

$$p_\lambda^{(1)} = e^{-2n\left(\alpha_1 - \widehat{R}_1(\lambda)\right)_+^2} \text{ and } p_\lambda^{(2)} = e^{-2n\left(\alpha_2 - \widehat{R}_2(\lambda)\right)_+^2},$$

where

$$\widehat{R}_1(\lambda) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\left\{\mathcal{T}_\lambda(X_i) = \emptyset\right\}$$

and

$$\widehat{R}_2(\lambda) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\left\{Y_i \notin \mathcal{T}_\lambda(X_i), \mathcal{T}_\lambda(X_i) \neq \emptyset\right\} - \alpha_2 \mathbb{1}\left\{\mathcal{T}_\lambda(X_i) = \emptyset\right\}.$$ </>

</> The second empirical risk, $\widehat{R}_2$, looks different from a standard empirical risk

```
# ood is an OOD detector, model is classifier with softmax output
lambda1s = torch.linspace(0,1,N) # Usually N ~= 1000
lambda2s = torch.linspace(0,1,N)
losses = torch.zeros((2,n,N,N)) # 2 losses, n data points, N x N lambdas
# The following loop can be massively parallelized (and GPU accelerated)
for (i,j,k) in [
                    (i,j,k) for i in range(n)
                            for j in range(N)
                            for k in range(N)
               ]:
  softmaxes = model(X[i].unsqueeze(0)).softmax(1).squeeze()
  cumsum = softmaxes.sort(descending=True)[0].cumsum(0)[Y[i]]
  if odd(X) > lambda1s[j]:
    losses[0,i,j,k] = 1
    continue
  losses[1,i,j,k] = int(cumsum > lambda2s[k])
risks = losses.mean(dim=1) # 2 x N x N
risks[1] = risks[1] - alpha2*risks[0]
pval1s = torch.exp(
           -2*n*(torch.relu(alpha1-risks[0])**2)
         ) # Any p-value
pval2s = torch.exp(
           -2*n*(torch.relu(alpha2-risks[1])**2)
         ) # Any p-value
pvals = torch.maximum(pval1s,pval2s)
# Can replace Bonferroni with SGT as in LTT paper
valid = torch.where(pvals <= delta/(N*N))
lambda_hat = [lambda1s[valid[0]], lambda2s[valid[1]]]
```

Figure B.4: PyTorch code for simultaneously controlling the type-1 error of OOD detection and prediction set coverage.

Since the maximum of two p-values is also a p-value (you can check this manually by verifying its super-uniformity), we can form the p-value for our union null as

$$p_\lambda = \max\left(p_\lambda^{(1)}, p_\lambda^{(2)}\right).$$

In practice, as before, we use the p-values from the HB inequality as opposed to those from Hoeffding. Then, instead of Bonferroni correction, we combine them with a less conservative form of sequential graphical testing; see Angelopoulos *et al.* [2] for these more mathematical details.

because of the conditioning. In other words, not all of our calibration data points have nonempty prediction sets; see Section 4 of Angelopoulos *et al.* [2] to learn more about this point.

For the purposes of this development, it suffices to return the Bonferroni region,

$$\widehat{\Lambda} = \left\{ \lambda : p_\lambda \leq \frac{\delta}{|\Lambda|} \right\}.$$

Then, every element of $\widehat{\Lambda}$ controls both risks simultaneously. See Figure B.4 for a PyTorch implementation of this procedure.

C

Concentration Properties of the Empirical Coverage

We adopt the same notation as Section 3.

The variation in $\overline{C}$ has three components. First, n is finite. We analyzed how this leads to fluctuations in the coverage in Section 3.2. The second source of fluctuations is the finiteness of n_{val}, the size of the validation set. A small number of validation points can result in a high-variance estimate of the coverage. This makes the histogram of the C_j wider than the beta distribution above. However, as we will now show, C_j has an analytical distribution that allows us to exactly understand the histogram's expected properties.

We now examine the distribution of C_j. Because C_j is an average of indicator functions, it looks like it is a binomially distributed random variable. This is true conditionally on the calibration data, but not marginally. This is because the mean of the binomial is beta distributed; as we showed in the above analysis, $\mathbb{E}\left[C_j \middle| \{(X_{i,j}, Y_{i,j})\}_{i=1}^{n}\right] \sim \text{Beta}(n + 1 - l, l)$, where $(X_{i,j}, Y_{i,j})$ is the ith calibration point in the jth trial. Conveniently, binomial random variables with beta-distributed mean,

$$C_j \sim \frac{1}{n_{\text{val}}}\text{Binom}(n_{\text{val}}, \mu) \text{ where } \mu \sim \text{Beta}(n + 1 - l, l),$$

are called *beta-binomial* random variables. We refer to this distribution as $\text{BetaBinom}(n_{\text{val}}, n+1-l, l)$; its properties, such as moments and probability mass function, can be found in standard references.

Knowing the analytic form of the C_j allows us to directly plot its distribution. After a sufficient number of trials R, the histogram of C_j should converge almost exactly to its analytical PMF (which is only a function of α, n, and n_{val}). The plot in Figure C.1 shows how the histograms should look with different values of n_{val} and large R. Code for producing these plots is also available in the aforementioned Jupyter notebook.

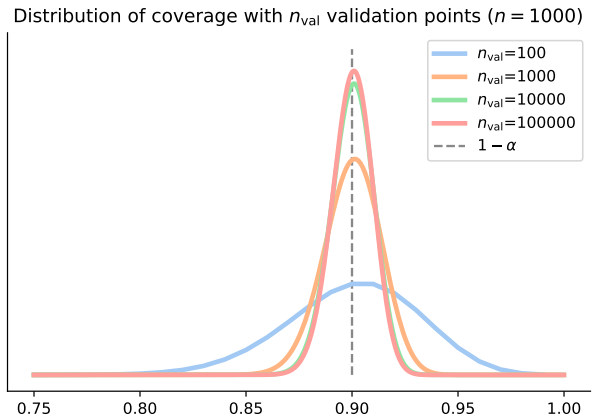


Figure C.1: The distribution of empirical coverage converges to the Beta distribution in Figure 3.3 as n_{val} grows. However, for small values of n_{val}, the histogram can have an inflated variance.

The final source of fluctuations is due to the finite number of experiments, R. We have now shown that the C_j are independent beta-binomial random variables. Unfortunately, the distribution of $\overline{C}$—the mean of R independent beta-binomial random variables—does not have a closed form. However, we can simulate the distribution easily, and we visualize it for several realistic choices of R, n_{val}, and n in Figure C.2.

Furthermore, we can analytically reason about the tail properties of $\overline{C}$. Since $\overline{C}$ is the average of R i.i.d. beta-binomial random variables, its mean and standard deviation are

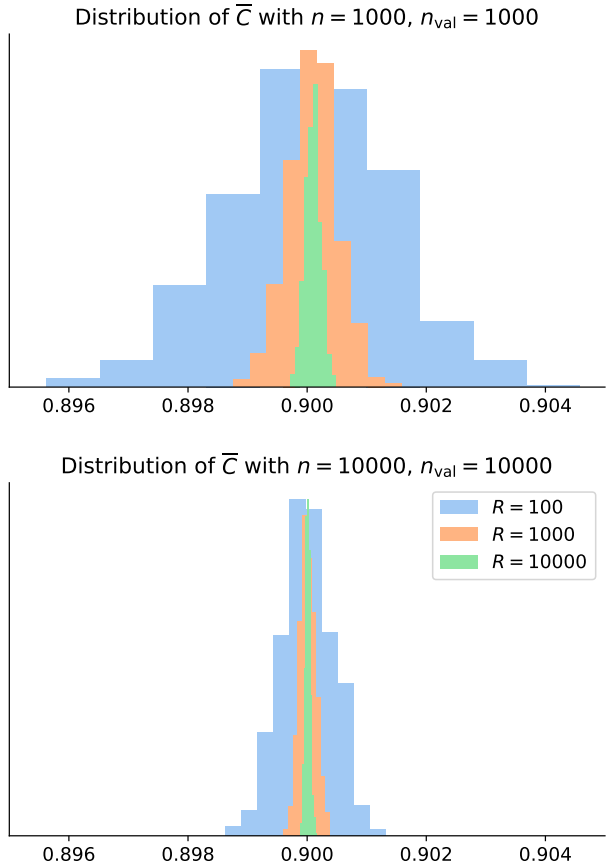


Figure C.2: The distribution of average empirical coverage over R trials with n calibration points and n_{val} validation points.[</>]

$$\mathbb{E}\left(\overline{C}\right) = 1 - \frac{l}{n+1}$$

and

$$\sqrt{\text{Var}\left(\overline{C}\right)} = \sqrt{\frac{l(n+1-l)(n+n_{\text{val}}+1)}{n_{\text{val}}R(n+1)^2(n+2)}} = \mathcal{O}\left(\frac{1}{\sqrt{R\min(n, n_{\text{val}})}}\right).$$

[</>] https://github.com/aangelopoulos/conformal-prediction/blob/main/notebooks/correctness_checks.ipynb

The best way for a practitioner to carefully debug their procedure is to compute $\overline{C}$ empirically, and then cross-reference with Figure C.2. We give code to simulate histograms with any n, R, and n_{val} in the linked notebook of Figure C.2. If the simulated average empirical coverage does not align well with the coverage observed on the real data, there is likely a problem in the conformal implementation.

D

Theorem and Proof: Coverage Property of Conformal Prediction

This is a standard proof of validity for split-conformal prediction first appearing in Papadopoulos *et al.* [88], but we reproduce it here for completeness. Let us begin with the lower bound.

Theorem D.1 (Conformal calibration coverage guarantee)**.** Suppose $(X_i, Y_i)_{i=1,\ldots,n}$ and $(X_{\text{test}}, Y_{\text{test}})$ are i.i.d. Then define $\hat{q}$ as

$$\hat{q} = \inf\left\{q : \frac{|\{i : s(X_i, Y_i) \leq q\}|}{n} \geq \frac{\lceil (n+1)(1-\alpha) \rceil}{n}\right\}.$$

and the resulting prediction sets as

$$\mathcal{C}(X) = \{y : s(X, y) \leq \hat{q}\}.$$

Then,

$$P\Big(Y_{\text{test}} \in \mathcal{C}(X_{\text{test}})\Big) \geq 1 - \alpha.$$

This is the same coverage property as (1.1) in the introduction, but written more formally. As a technical remark, the theorem also holds if the observations to satisfy the weaker condition of exchangeability; see Vovk *et al.* [116]. Below, we prove the lower bound.

Proof of Theorem 1.1. Let $s_i = s(X_i, Y_i)$ for $i = 1, \ldots, n$ and $s_{\text{test}} = s(X_{\text{test}}, Y_{\text{test}})$. To avoid handling ties, we consider the case where the s_i

are distinct with probability 1. See Tibshirani *et al.* [107] for a proof in the general case.

Without loss of generality we assume the calibration scores are sorted so that $s_1 < \cdots < s_n$. In this case, we have that $\hat{q} = s_{\lceil (n+1)(1-\alpha) \rceil}$ when $\alpha \geq \frac{1}{n+1}$ and $\hat{q} = \infty$ otherwise. Note that in the case $\hat{q} = \infty$, $\mathcal{C}(X_{\text{test}}) = \mathcal{Y}$, so the coverage property is trivially satisfied; thus, we only have to handle the case when $\alpha \geq \frac{1}{n+1}$. We proceed by noticing the equality of the two events

$$\{Y_{\text{test}} \in \mathcal{C}(X_{\text{test}})\} = \{s_{\text{test}} \leq \hat{q}\}.$$

Combining this with the definition of $\hat{q}$ yields

$$\{Y_{\text{test}} \in \mathcal{C}(X_{\text{test}})\} = \{s_{\text{test}} \leq s_{\lceil (n+1)(1-\alpha) \rceil}\}.$$

Now comes the crucial insight. By exchangeability of the variables $(X_1, Y_1), \ldots, (X_{\text{test}}, Y_{\text{test}})$, we have

$$P(s_{\text{test}} \leq s_k) = \frac{k}{n+1}$$

for any integer k. In words, s_{test} is equally likely to fall in anywhere between the calibration points $s_1, \ldots, s_n$. Note that above, the randomness is over all variables $s_1, \ldots, s_n, s_{\text{test}}$

From here, we conclude

$$P(s_{\text{test}} \leq s_{\lceil (n+1)(1-\alpha)) \rceil}) = \frac{\lceil (n+1)(1-\alpha) \rceil}{(n+1)} \geq 1 - \alpha,$$

which implies the desired result. □

Now we will discuss the upper bound. Technically, the upper bound only holds when the distribution of the conformal score is continuous, avoiding ties. In practice, however, this condition is not important, because the user can always add a vanishing amount of random noise to the score. We will state the theorem now, and defer its proof.

Theorem D.2 (Conformal calibration upper bound)**.** Additionally, if the scores $s_1, ..., s_n$ have a continuous joint distribution, then

$$P\Big(Y_{\text{test}} \in \mathcal{C}(X_{\text{test}}, U_{\text{test}}, \hat{q})\Big) \leq 1 - \alpha + \frac{1}{n+1}.$$

Proof. See Theorem 2.2 of Lei *et al.* [68]. □

References

[1] D. J. Aldous, "Exchangeability and related topics," in *École d'Été de Probabilités de Saint-Flour XIII—1983*, 1985, pp. 1–198.

[2] A. N. Angelopoulos, S. Bates, E. J. Candès, M. I. Jordan, and L. Lei, "Learn then test: Calibrating predictive algorithms to achieve risk control," 2021. arXiv: 2110.01052.

[3] A. N. Angelopoulos, S. Bates, A. Fisch, L. Lei, and T. Schuster, "Conformal risk control," 2022. arXiv: 2208.02814.

[4] A. N. Angelopoulos, S. Bates, T. Zrnic, and M. I. Jordan, "Private prediction sets," 2021. arXiv: 2102.06202.

[5] A. N. Angelopoulos, A. P. Kohli, S. Bates, M. I. Jordan, J. Malik, T. Alshaabi, S. Upadhyayula, and Y. Romano, "Image-to-image regression with distribution-free uncertainty quantification and applications in imaging," 2022. arXiv: 2202.05265.

[6] A. N. Angelopoulos, S. Bates, J. Malik, and M. I. Jordan, "Uncertainty sets for image classifiers using conformal prediction," in *International Conference on Learning Representations*, 2021. URL: https://openreview.net/forum?id=eNdiU_DbM9.

[7] R. F. Barber, E. J. Candes, A. Ramdas, and R. J. Tibshirani, "Predictive inference with the jackknife+," *The Annals of Statistics*, vol. 49, no. 1, 2021, pp. 486–507.

[8] R. F. Barber, E. J. Candes, A. Ramdas, and R. J. Tibshirani, "Conformal prediction beyond exchangeability," 2022. arXiv: 2202.13415.

[9] O. Bastani, V. Gupta, C. Jung, G. Noarov, R. Ramalingam, and A. Roth, "Practical adversarial multivalid conformal prediction," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022. URL: https://openreview.net/forum?id=QNjyrDBx6tz.

[10] S. Bates, A. Angelopoulos, L. Lei, J. Malik, and M. Jordan, "Distribution-free, risk-controlling prediction sets," *Journal of the Association for Computing Machinery*, vol. 68, no. 6, Sep. 2021.

[11] S. Bates, E. Candès, L. Lei, Y. Romano, and M. Sesia, "Testing for outliers with conformal p-values," 2021. arXiv: 2104.08279.

[12] H. Boström, H. Linusson, T. Löfström, and U. Johansson, "Accelerating difficulty estimation for conformal regression forests," *Annals of Mathematics and Artificial Intelligence*, vol. 81, no. 1, 2017, pp. 125–144.

[13] F. Bretz, W. Maurer, W. Brannath, and M. Posch, "A graphical approach to sequentially rejective multiple test procedures," *Statistics in Medicine*, vol. 28, no. 4, 2009, pp. 586–604.

[14] E. J. Candès, L. Lei, and Z. Ren, "Conformalized survival analysis," 2021. arXiv: 2103.09763.

[15] M. Cauchois, S. Gupta, A. Ali, and J. C. Duchi, "Robust validation: Confident predictions even when distributions shift," 2020. arXiv: 2008.04267.

[16] M. Cauchois, S. Gupta, and J. Duchi, "Knowing what you know: Valid and validated confidence sets in multiclass and multilabel prediction," 2020. arXiv: 2004.10181.

[17] S. Chatterjee and P. Qiu, "Distribution-free cumulative sum control charts using bootstrap-based control limits," *The Annals of Applied Statistics*, vol. 3, no. 1, 2009, pp. 349–369.

[18] P. Chaudhuri, "Global nonparametric estimation of conditional quantile functions and their derivatives," *Journal of Multivariate Analysis*, vol. 39, no. 2, 1991, pp. 246–269.

[19] J. Cherian and L. Bronner, "How the Washington Post estimates outstanding votes for the 2020 presidential election," *Washington Post*, 2021. URL: https://s3.us-east-1.amazonaws.com/elex-models-prod/2020-general/write-up/election_model_writeup.pdf.

[20] V. Chernozhukov, K. Wüthrich, and Z. Yinchu, "Exact and robust conformal inference methods for predictive machine learning with dependent data," in *Conference On Learning Theory*, PMLR, pp. 732–749, 2018.

[21] V. Chernozhukov, K. Wüthrich, and Y. Zhu, "An exact and robust conformal inference method for counterfactual and synthetic controls," *Journal of the American Statistical Association*, 2021, pp. 1–16.

[22] E. Chung and J. P. Romano, "Exact and asymptotically robust permutation tests," *The Annals of Statistics*, vol. 41, no. 2, 2013, pp. 484–507.

[23] A. Church, "On the concept of a random sequence," *Bulletin of the American Mathematical Society*, vol. 46, no. 2, 1940, pp. 130–135.

[24] B. De Finetti, "Funzione caratteristica di un fenomeno aleatorio," in *Atti del Congresso Internazionale dei Matematici: Bologna del 3 al 10 de Settembre di 1928*, pp. 179–190, 1929.

[25] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018. arXiv: 1810.04805.

[26] P. Diaconis and D. Freedman, "Finite exchangeable sequences," *The Annals of Probability*, 1980, pp. 745–764.

[27] A. Dixit, L. Lindemann, M. Cleaveland, S. Wei, G. J. Pappas, and J. W. Burdick, "Adaptive conformal prediction for motion planning among dynamic agents," 2022. arXiv: 2212.00278.

[28] E. Dobriban, *Topics in Modern Statistical Learning (STAT 991, UPenn, 2022 Spring)*, Dec. 2022. URL: https://github.com/dobriban/Topics-In-Modern-Statistical-Learning.

[29] A. V. Dorogush, V. Ershov, and A. Gulin, "Catboost: Gradient boosting with categorical features support," 2018. arXiv: 1810.11363.

[30] R. Dunn, L. Wasserman, and A. Ramdas, "Distribution-free prediction sets with random effects," 2018. arXiv: 1809.07441.

[31] B. Efron and R. J. Tibshirani, *An introduction to the bootstrap*. CRC press, 1994.

[32] G. T. Fechner, *Kollektivmasslehre*. Engelmann, 1897.

[33] S. Feldman, S. Bates, and Y. Romano, "Improving conditional coverage via orthogonal quantile regression," in *Advances in Neural Information Processing Systems*, 2021.

[34] A. Fisch, T. Schuster, T. Jaakkola, and D. Barzilay, "Few-shot conformal prediction with auxiliary tasks," in *International Conference on Machine Learning*, vol. 139, pp. 3329–3339, 2021.

[35] A. Fisch, T. Schuster, T. S. Jaakkola, and R. Barzilay, "Efficient conformal prediction via cascaded inference with expanded admission," in *International Conference on Learning Representations*, 2021.

[36] R. A. Fisher, "Design of experiments," *British Medical Journal*, vol. 1, no. 3923, 1936, p. 554.

[37] R. Foygel Barber, E. J. Candes, A. Ramdas, and R. J. Tibshirani, "The limits of distribution-free conditional predictive inference," *Information and Inference: A Journal of the IMA*, vol. 10, no. 2, 2021, pp. 455–482.

[38] D. A. Freedman, "Bernard Friedman's urn," *The Annals of Mathematical Statistics*, 1965, pp. 956–970.

[39] A. Gammerman, V. Vovk, and V. Vapnik, "Learning by transduction," *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, vol. 14, 1998, pp. 148–155.

[40] I. Gibbs and E. Candès, "Adaptive conformal inference under distribution shift," 2021. arXiv: 2106.00170.

[41] I. Gibbs and E. Candès, "Conformal inference for online prediction with arbitrary distribution shifts," 2022. arXiv: 2208.08401.

[42] L. Guan, "Conformal prediction with localization," 2020. arXiv: 1908.08558.

[43] L. Guan and R. Tibshirani, "Prediction and outlier detection in classification problems," 2019. arXiv: 1905.04396.

[44] C. Gupta, A. K. Kuchibhotla, and A. Ramdas, "Nested conformal prediction and quantile out-of-bag ensemble methods," *Pattern Recognition*, 2021, p. 108 496.

[45] C. Gupta and A. Ramdas, "Distribution-free calibration guarantees for histogram binning without sample splitting," in *International Conference on Machine Learning*, vol. 139, pp. 3942–3952, 2021.

[46] L. Hanu and Unitary team, *Detoxify*, 2020. URL: https://github.com/unitaryai/detoxify.

[47] Y. Hechtlinger, B. Poczos, and L. Wasserman, "Cautious deep learning," 2018. arXiv: 1805.09460.

[48] E. Hewitt and L. J. Savage, "Symmetric measures on Cartesian products," *Transactions of the American Mathematical Society*, vol. 80, no. 2, 1955, pp. 470–501.

[49] P. Hoff, "Bayes-optimal prediction with frequentist coverage control," 2021. arXiv: 2105.14045.

[50] X. Hu and J. Lei, "A distribution-free test of covariate shift using conformal prediction," 2020. arXiv: 2010.07147.

[51] R. Izbicki, G. Shimizu, and R. Stern, "Flexible distribution-free conditional predictive bands using density estimators," in *Proceedings of Machine Learning Research*, vol. 108, pp. 3068–3077, PMLR, 2020.

[52] U. Johansson, H. Boström, T. Löfström, and H. Linusson, "Regression conformal prediction with random forests," *Machine learning*, vol. 97, no. 1, 2014, pp. 155–176.

[53] C. Jung, G. Noarov, R. Ramalingam, and A. Roth, "Batch multivalid conformal prediction," 2022. arXiv: 2209.15145.

[54] J. F. Kingman, "Uses of exchangeability," *The Annals of Probability*, vol. 6, no. 2, 1978, pp. 183–197.

[55] R. Koenker, *Quantile Regression*. Cambridge University Press, 2005.

[56] R. Koenker, "Additive models for quantile regression: Model selection and confidence bandaids," *Brazilian Journal of Probability and Statistics*, vol. 25, no. 3, 2011, pp. 239–262.

[57] R. Koenker and G. Bassett Jr, "Regression quantiles," *Econometrica: Journal of the Econometric Society*, vol. 46, no. 1, 1978, pp. 33–50.

[58] R. Koenker, V. Chernozhukov, X. He, and L. Peng, "Handbook of quantile regression," 2018.

[59] P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, *et al.*, "Wilds: A benchmark of in-the-wild distribution shifts," in *International Conference on Machine Learning*, PMLR, pp. 5637–5664, 2021.

[60] A. Kolmogorov, "Logical basis for information theory and probability theory," *IEEE Transactions on Information Theory*, vol. 14, no. 5, 1968, pp. 662–664.

[61] A. N. Kolmogorov, "Three approaches to the quantitative definition of information," *Problems of Information Transmission*, vol. 1, no. 1, 1965, pp. 1–7.

[62] A. N. Kolmogorov, "Combinatorial foundations of information theory and the calculus of probabilities," *Russian Mathematical Surveys*, vol. 38, no. 4, 1983, pp. 29–40.

[63] A. K. Kuchibhotla and R. A. Berk, "Nested conformal prediction sets for classification with applications to probation data," 2021. arXiv: 2104.09358.

[64] Y. Lee and R. F. Barber, "Distribution-free inference for regression: Discrete, continuous, and in between," 2021. arXiv: 2105.14075.

[65] E. L. Lehmann, "The power of rank tests," *The Annals of Mathematical Statistics*, 1953, pp. 23–43.

[66] J. Lei, "Classification with confidence," *Biometrika*, vol. 101, no. 4, Oct. 2014, pp. 755–769. DOI: 10.1093/biomet/asu038.

[67] J. Lei, "Fast exact conformalization of the lasso using piecewise linear homotopy," *Biometrika*, vol. 106, no. 4, 2019, pp. 749–764.

[68] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman, "Distribution-free predictive inference for regression," *Journal of the American Statistical Association*, vol. 113, no. 523, 2018, pp. 1094–1111. DOI: 10.1080/01621459.2017.1307116..

[69] J. Lei, A. Rinaldo, and L. Wasserman, "A conformal prediction approach to explore functional data," *Annals of Mathematics and Artificial Intelligence*, vol. 74, 2015, pp. 29–43. DOI: 10.1007/s10472-013-9366-6.

[70] J. Lei, J. Robins, and L. Wasserman, "Efficient nonparametric conformal prediction regions," 2011. arXiv: 1111.1418.

[71] J. Lei, J. Robins, and L. Wasserman, "Distribution-free prediction sets," *Journal of the American Statistical Association*, vol. 108, no. 501, 2013, pp. 278–287.

[72] J. Lei and L. Wasserman, "Distribution-free prediction bands for non-parametric regression," *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, 2014, pp. 71–96.

[73] L. Lei and E. J. Candès, "Conformal inference of counterfactuals and individual treatment effects," 2020. arXiv: 2006.06138.

[74] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European conference on computer vision*, Springer, pp. 740–755, 2014.

[75] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe planning in dynamic environments using conformal prediction," 2022. arXiv: 2210.10254.

[76] H. Linusson, U. Norinder, H. Boström, U. Johansson, and T. Löfström, "On the calibration of aggregated conformal predictors," in *Conformal and probabilistic prediction and applications*, PMLR, pp. 154–173, 2017.

[77] C. Lu and J. Kalpathy-Cramer, "Distribution-free federated learning with conformal predictions," 2021. arXiv: 2110.07661.

[78] C. Lu, A. Lemay, K. Chang, K. Hoebel, and J. Kalpathy-Cramer, "Fair conformal predictors for applications in medical imaging," 2021. arXiv: 2109.04392.

[79] A. Malinin, N. Band, G. Chesnokov, Y. Gal, M. J. Gales, A. Noskov, A. Ploskonosov, L. Prokhorenkova, I. Provilkov, V. Raina, *et al.*, "Shifts: A dataset of real distributional shift across multiple large-scale tasks," 2021. arXiv: 2107.07455.

[80] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The Annals of Mathematical Statistics*, 1947, pp. 50–60.

[81] V. Manokhin, *Awesome Conformal Prediction*, version v1.0.0, Apr. 2022. DOI: 10.5281/zenodo.6467205.

[82] T. Melluish, C. Saunders, I. Nouretdinov, and V. Vovk, "Comparing the bayes and typicalness frameworks," in *European Conference on Machine Learning*, Springer, pp. 360–371, 2001.

[83] R. von Mises, "Grundlagen der wahrscheinlichkeitsrechnung," *Mathematische Zeitschrift*, vol. 5, no. 1, 1919, pp. 52–99.

[84] F. Mota, S. Aaronson, L. Antunes, and A. Souto, "Sophistication as randomness deficiency," in *International Workshop on Descriptional Complexity of Formal Systems*, Springer, pp. 172–181, 2013.

[85] E. Ndiaye and I. Takeuchi, "Computing full conformal prediction set with approximate homotopy," in *Advances in Neural Information Processing Systems*, 2019. URL: https://arxiv.org/pdf/1909.09365.pdf.

[86] E. Ndiaye and I. Takeuchi, "Root-finding approaches for computing conformal prediction set," *Machine Learning*, 2022. DOI: 10.1007/s10994-022-06233-5.

[87] R. I. Oliveira, P. Orenstein, T. Ramos, and J. V. Romano, "Split conformal prediction for dependent data," 2022. arXiv: 2203.15885.

[88] H. Papadopoulos, K. Proedrou, V. Vovk, and A. Gammerman, "Inductive confidence machines for regression," in *Machine Learning: European Conference on Machine Learning*, pp. 345–356, 2002.

[89] S. Park, S. Li, O. Bastani, and I. Lee, "PAC confidence predictions for deep neural network classifiers," in *International Conference on Learning Representations*, 2021. URL: https://openreview.net/forum?id=Qk-Wq5AIjpq.

[90] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, 2014, pp. 215–249. DOI: https://doi.org/10.1016/j.sigpro.2013.12.026.

[91] E. J. Pitman, "Significance tests which may be applied to samples from any populations," *Supplement to the Journal of the Royal Statistical Society*, vol. 4, no. 1, 1937, pp. 119–130.

[92] B. Póczos, A. Singh, A. Rinaldo, and L. Wasserman, "Distribution-free distribution regression," in *Artificial Intelligence and Statistics*, PMLR, pp. 507–515, 2013.

[93] A. Podkopaev and A. Ramdas, "Tracking the risk of a deployed model and detecting harmful distribution shifts," 2021. arXiv: 2110.06177.

[94] C. P. Porter, "Kolmogorov on the role of randomness in probability theory," *Mathematical Structures in Computer Science*, vol. 24, no. 3, 2014.

[95] Y. Romano, R. F. Barber, C. Sabatti, and E. Candès, "With malice toward none: Assessing uncertainty via equalized coverage," *Harvard Data Science Review*, vol. 2, no. 2, Apr. 30, 2020. DOI: 10.1162/99608f92.03f00592.

[96] Y. Romano, E. Patterson, and E. Candès, "Conformalized quantile regression," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 3543–3553.

[97] Y. Romano, M. Sesia, and E. J. Candès, "Classification with valid and adaptive coverage," 2020. arXiv: 2006.02544.

[98] M. Sadinle, J. Lei, and L. Wasserman, "Least ambiguous set-valued classifiers with bounded error levels," *Journal of the American Statistical Association*, vol. 114, 2019, pp. 223–234.

[99] C. Saunders, A. Gammerman, and V. Vovk, "Transduction with confidence and credibility," 1999.

[100] T. Schuster, A. Fisch, T. Jaakkola, and R. Barzilay, "Consistent accelerated inference via confident adaptive transformers," *Empirical Methods in Natural Language Processing*, 2021.

[101] G. Shafer and V. Vovk, "The sources of Kolmogorov's Grundbegriffe," *Statistical Science*, vol. 21, no. 1, 2006, pp. 70–98.

[102] G. Shafer and V. Vovk, "A tutorial on conformal prediction," *Journal of Machine Learning Research*, vol. 9, no. Mar, 2008, pp. 371–421.

[103] Z. Sidak, P. K. Sen, and J. Hajek, *Theory of rank tests*. Elsevier, 1999.

[104] I. Steinwart and A. Christmann, "Estimating conditional quantiles with the help of the pinball loss," *Bernoulli*, vol. 17, no. 1, 2011, pp. 211–225.

[105] D. Stutz, K. D. Dvijotham, A. T. Cemgil, and A. Doucet, "Learning optimal conformal classifiers," in *International Conference on Learning Representations*, 2022.

[106] I. Takeuchi, Q. V. Le, T. D. Sears, and A. J. Smola, "Nonparametric quantile estimation," *Journal of Machine Learning Research*, vol. 7, 2006, pp. 1231–1264.

[107] R. J. Tibshirani, R. Foygel Barber, E. Candes, and A. Ramdas, "Conformal prediction under covariate shift," in *Advances in Neural Information Processing Systems 32*, 2019, pp. 2530–2540.

[108] J. W. Tukey, "Non-parametric estimation II. Statistically equivalent blocks and tolerance regions–the continuous case," *Annals of Mathematical Statistics*, vol. 18, no. 4, 1947, pp. 529–539. DOI: 10.1214/aoms/1177730343.

[109] J. Ville, "Etude critique de la notion de collectif," *Bull. Amer. Math. Soc*, vol. 45, no. 11, 1939, p. 824.

[110] D. Volkhonskiy, E. Burnaev, I. Nouretdinov, A. Gammerman, and V. Vovk, "Inductive conformal martingales for change-point detection," in *Conformal and Probabilistic Prediction and Applications*, PMLR, pp. 132–153, 2017.

[111] V. Vovk, "Kolmogorov's complexity conception of probability," *Synthese Library*, 2001, pp. 51–70.

[112] V. Vovk, "On-line confidence machines are well-calibrated," in *The 43rd Annual IEEE Symposium on Foundations of Computer Science*, IEEE, pp. 187–196, 2002.

[113] V. Vovk, "Conditional validity of inductive conformal predictors," in *Proceedings of the Asian Conference on Machine Learning*, vol. 25, pp. 475–490, 2012.

[114] V. Vovk, "Cross-conformal predictors," *Annals of Mathematics and Artificial Intelligence*, vol. 74, no. 1-2, 2015, pp. 9–28.

[115] V. Vovk, "Testing randomness online," *Statistical Science*, vol. 36, no. 4, 2021, pp. 595–611.

[116] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic Learning in a Random World*. Springer, 2005. DOI: 10.1007/b106715..

[117] V. Vovk, A. Gammerman, and C. Saunders, "Machine-learning applications of algorithmic randomness," in *International Conference on Machine Learning*, pp. 444–453, 1999.

[118] V. Vovk, I. Nouretdinov, and A. Gammerman, "Testing exchangeability on-line," in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 768–775, 2003.

[119] V. Vovk and I. Petej, "Venn-Abers predictors," 2012. arXiv: 1211.0025.

[120] V. Vovk, G. Shafer, and I. Nouretdinov, "Self-calibrating probability forecasting.," in *Neural Information Processing Systems*, pp. 1133–1140, 2003.

[121] V. Vovk, J. Shen, V. Manokhin, and M.-g. Xie, "Nonparametric predictive distributions based on conformal prediction," *Machine Learning*, 2017, pp. 1–30.

[122] V. G. Vovk, "On the concept of the Bernoulli property," *Russian Mathematical Surveys*, vol. 41, no. 1, 1986, p. 247.

[123] A. Wald, "Die widerspruchfreiheit des kollectivbegriffes der wahrscheinlichkeitsrechnung," *Ergebnisse Eines Mathematischen Kolloquiums*, vol. 8, no. 38-72, 1937, p. 37.

[124] A. Wald, "An extension of Wilks' method for setting tolerance limits," *Annals of Mathematical Statistics*, vol. 14, no. 1, 1943, pp. 45–55. DOI: 10.1214/aoms/1177731491.

[125] L. Wasserman, "Frasian inference," *Statistical Science*, vol. 26, no. 3, 2011, pp. 322–325.

[126] S. S. Wilks, "Determination of sample sizes for setting tolerance limits," *Annals of Mathematical Statistics*, vol. 12, no. 1, 1941, pp. 91–96. DOI: 10.1214/aoms/1177731788.

[127] S. S. Wilks, "Statistical prediction with special reference to the problem of tolerance limits," *Annals of Mathematical Statistics*, vol. 13, no. 4, 1942, pp. 400–409. DOI: 10.1214/aoms/1177731537.

[128] C. Xu and Y. Xie, "Conformal prediction interval for dynamic time-series," in *International Conference on Machine Learning*, PMLR, pp. 11 559–11 569, 2021.

[129] M. Yin, C. Shi, Y. Wang, and D. M. Blei, "Conformal sensitivity analysis for individual treatment effects," 2021. arXiv: 2112.03493.

[130] M. Zaffran, O. Féron, Y. Goude, J. Josse, and A. Dieuleveut, "Adaptive conformal predictions for time series," in *International Conference on Machine Learning*, PMLR, pp. 25 834–25 866, 2022.

[131] K. Q. Zhou, S. L. Portnoy, *et al.*, "Direct use of regression quantiles to construct confidence sets in linear models," *The Annals of Statistics*, vol. 24, no. 1, 1996, pp. 287–306.

[132] K. Q. Zhou and S. L. Portnoy, "Statistical inference on heteroscedastic models based on regression quantiles," *Journal of Nonparametric Statistics*, vol. 9, no. 3, 1998, pp. 239–260.