Uncertainty-aware Classification: From Rejection to Set-valued Prediction

Eyke Hüllermeier [1], Titouan Lorieul[2] and Willem Waegeman[3]

[1]Institute of Informatics, LMU Munich, Germany
[2]Inria, LIRMM, Université de Montpellier, France
[3]Department of Data Analysis and Mathematical Modelling, Ghent University, Belgium

Virtual Tutorial at ECML/PKDD 2021

Introduction

- Machine learning is inseparably connected with **uncertainty**.
- **Learning** in the sense of generalizing beyond the data seen so far is necessarily based on a process of **induction**.
- **Models** induced from data are never provably correct, but only hypothetical and therefore uncertain, and the same holds true for the **predictions** produced by a model.
- Other sources of uncertainty exist: incorrect **model assumptions**, noisy or imprecise **data**, etc.
- **Trustworthy representation** of uncertainty is desirable and should be considered as a key feature of any machine learning method, all the more in safety-critical application domains.

Introduction

- Many applications require safe and reliable predictions, and hence a certain level of **self-awareness** of ML systems:
  - equip predictions with an appropriate **quantification of uncertainty**,
  - **reject** a decision in cases of high uncertainty (abstention) ,
  - deliver a credible **set-valued prediction** (partial abstention),
  - ...

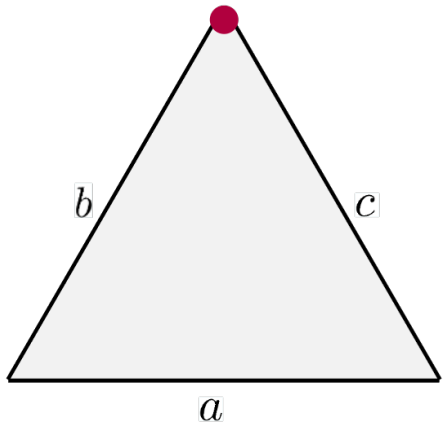
Driver assistance systems: a safety-critical application

Introduction

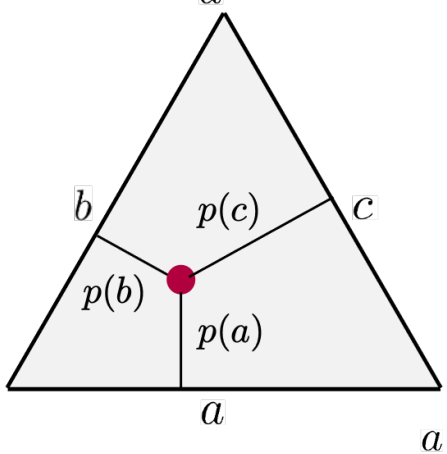
Example of a **lack of uncertainty-awareness**: Predictions by EfficientNet[1] on test images from ImageNet: For the left image, the neural network predicts "typewriter keyboard" with certainty 83.14 %, for the right image "stone wall" with certainty 87.63 %.

[1] M. Tan, Q.V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. ICML 2019.

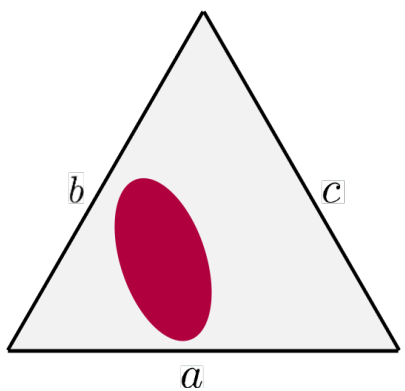
Levels of self-awareness and uncertainty representation



deterministic predictions,
no uncertainty represented



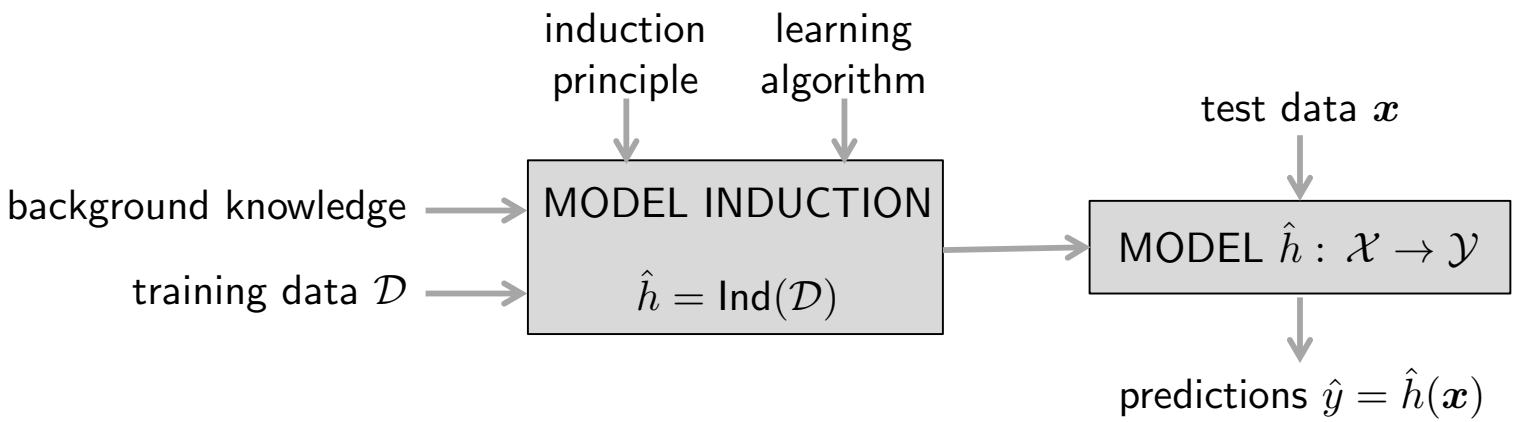
probabilistic predictions,
representation of aleatoric uncertainty



second-order uncertainty,
representation of epistemic uncertainty

Supervised learning and predictive uncertainty

- Uncertainty occurs in various facets in machine learning, and different **settings and learning problems** will usually require a different handling from an uncertainty modeling point of view.
- Here, we focus on the standard setting of **supervised learning** and **predictive uncertainty**.



- Assuming probabilistic data generation $p(\boldsymbol{x}, y) = p(\boldsymbol{x})p(y \mid \boldsymbol{x})$, **probabilistic predictors** (estimating $p(y \mid \boldsymbol{x})$) are natural primitives.

Supervised learning and predictive uncertainty

- A learner is given access to a set of (i.i.d.) **training data**

$$\mathcal{D} := \left\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\right\} \subset \mathcal{X} \times \mathcal{Y} \ ,$$

  where $\mathcal{X}$ is an instance space and $\mathcal{Y}$ the set of outcomes.
- Given a **hypothesis space** $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ and a loss function

$$\ell : \mathcal{Y} \times \mathcal{Y} \longrightarrow \mathbb{R} \, ,$$

  the goal of the learner is to induce a hypothesis $h^* \in \mathcal{H}$ with low risk

$$R(h) := \int_{\mathcal{X} \times \mathcal{Y}} \ell(h(\boldsymbol{x}), y) \, d \, P(\boldsymbol{x}, y) \ .$$

- The learner's choice is commonly guided by the **empirical risk**

$$R_{emp}(h) := \frac{1}{N} \sum_{i=1}^{N} \ell(h(\boldsymbol{x}_i), y_i) \ .$$

Supervised learning and predictive uncertainty

- Yet, since $R_{emp}(h)$ is only an estimation of the true risk $R(h)$, the **empirical risk minimizer**

$$\hat{h} := \arg\min_{h \in \mathcal{H}} R_{emp}(h)$$

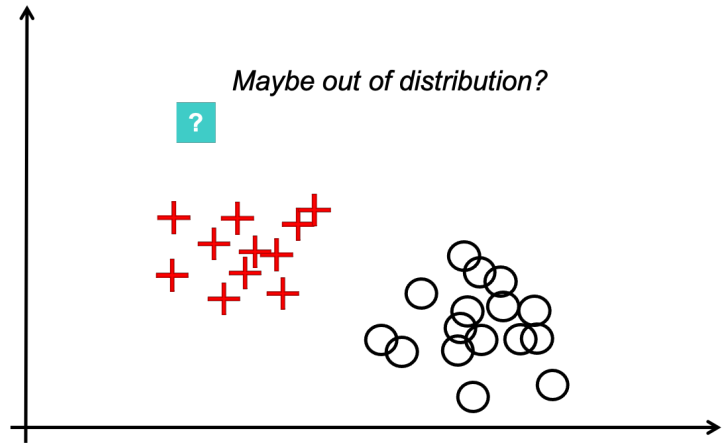
will normally not coincide with the true **risk minimizer**

$$h^* := \arg\min_{h \in \mathcal{H}} R(h) \,.$$

- Correspondingly, there remains **uncertainty** regarding h^* as well as the approximation quality of $\hat{h}$ (in the sense of its proximity to h^*) and its true risk $R(\hat{h})$.
- Eventually, one is often interested in the **predictive uncertainty**, i.e., the uncertainty related to the prediction $\hat{y}_q$ for an **individual** (query) instance $\boldsymbol{x}_q \in \mathcal{X}$.

Problem setting and assumptions

- A precise specification of the **problem setting** and **underlying assumptions** is an important prerequisite, not only for providing learning guarantees, but also for **uncertainty quantification**.

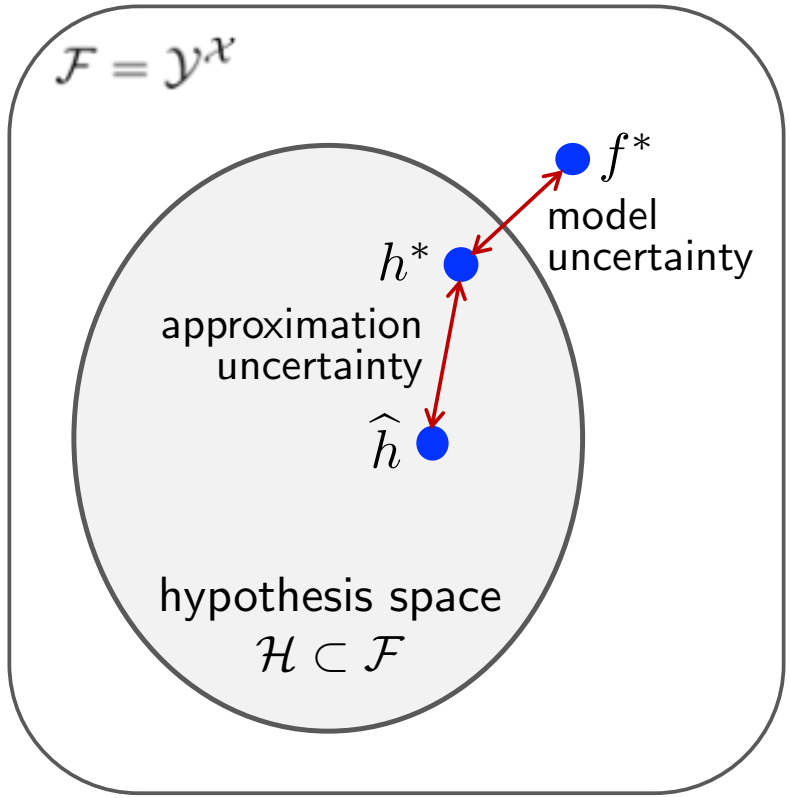


- Here, one might be quite sure about the class of the query under standard assumptions of binary classification, but much less so in a setting of **novelty detection**, where new classes may emerge.
- Likewise, assumptions such as **i.i.d. data generation** are really crucial (the past should be representative of the future).

Overview of this talk

1. Introduction
2. Uncertainty in classification: representation and methods (40 minutes)
3. Coffee break
4. Classification with reject option (25 minutes)
5. Set-valued prediction methods (25 minutes)
6. Coffee break
7. Rejection and set-valued prediction with distribution shift (40 minutes)

Sources of uncertainty



	point prediction	probability
ground truth	$f^*(\boldsymbol{x})$	$p(\cdot \mid \boldsymbol{x})$
best possible	$h^*(\boldsymbol{x})$	$p^*(\cdot \mid \boldsymbol{x}) = p(\cdot \mid \boldsymbol{x}, h^*)$
induced predictor	$\hat{h}(\boldsymbol{x})$	$\hat{p}(\cdot \mid \boldsymbol{x}) = p(\cdot \mid \boldsymbol{x}, \hat{h})$

Sources of uncertainty

- A query instance $\boldsymbol{x}_q$ gives rise to a conditional probability on $\mathcal{Y}$:

$$p(y \mid \boldsymbol{x}_q) = \frac{p(\boldsymbol{x}_q, y)}{p(\boldsymbol{x}_q)}$$

- Thus, even given full information in the form of the measure P (and its density p), uncertainty about the actual outcome y remains.
- The best point predictions (minimizing expected loss) are prescribed by the **pointwise Bayes predictor** f^*:

$$f^*(\boldsymbol{x}) := \operatorname*{arg\,min}_{\hat{y} \in \mathcal{Y}} \int_{\mathcal{Y}} \ell(y, \hat{y}) \, dP(y \mid \boldsymbol{x}) \,.$$

Sources of uncertainty

- The **Bayes predictor** does not necessarily coincide with the pointwise Bayes predictor.
- This discrepancy between h^* and f^* is connected to the uncertainty regarding the **right type of model** to be fit, and hence the choice of the hypothesis space $\mathcal{H}$.
- We shall refer to this uncertainty as **model uncertainty**.
- Due to model uncertainty, one cannot guarantee

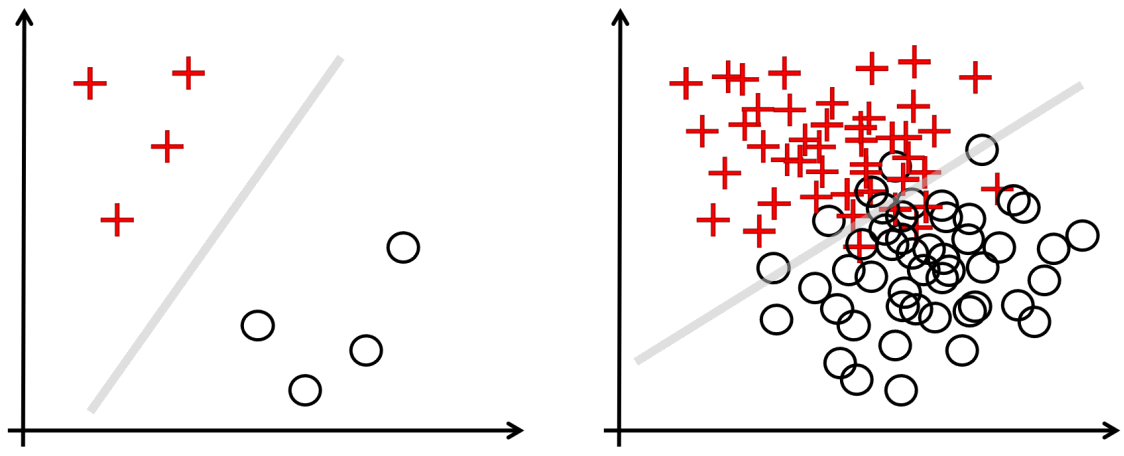
$$h^*(\boldsymbol{x}) = f^*(\boldsymbol{x})\,,$$

or, in the case of probabilistic predictions $p^*(y \mid \boldsymbol{x}) = p(y \mid \boldsymbol{x}, h^*)$, that

$$p^*(\cdot \mid \boldsymbol{x}) = p(\cdot \mid \boldsymbol{x})\,.$$

Sources of uncertainty

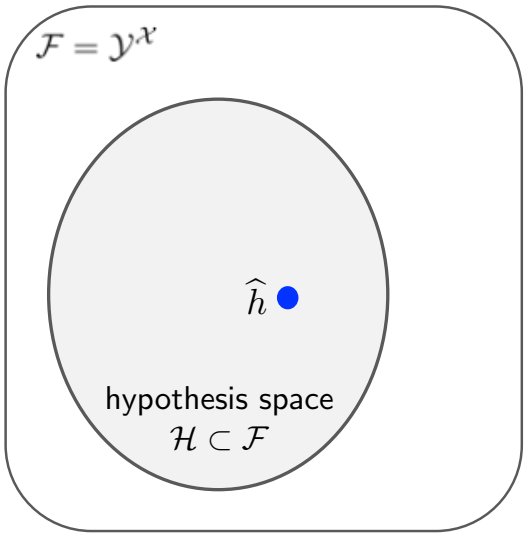
- Hypothesis $\hat{h}$ produced by the learner is an estimate of h^*.
- The quality of this estimate strongly depends on the quality and the amount of training data.

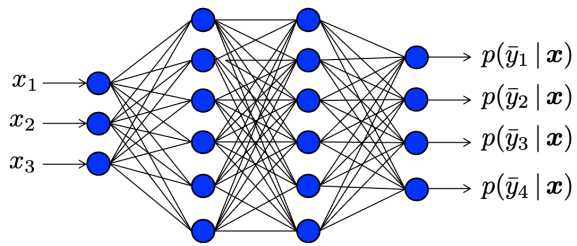


- We refer to the uncertainty about the discrepancy between $\hat{h}$ and h^* as **approximation uncertainty**.

Probabilistic models

- Probabilistic learners produce a single probabilistic predictor $\hat{h}$:





- Captures stochastic nature of dependence between instances and outcomes (aleatoric uncertainty).
- Yet, pretends full certainty about this dependence, thereby ignoring approximation and model uncertainty.

Bayesian learning

- In the Bayesian approach, learning corresponds to turning a **prior distribution** on $\mathcal{H}$ into a **posterior**:

$$p(h \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid h)\, p(h)}{p(\mathcal{D})} \propto p(\mathcal{D} \mid h)\, p(h)$$

- The **predictive posterior** distribution on $\mathcal{Y}$ is obtained via **model averaging**:

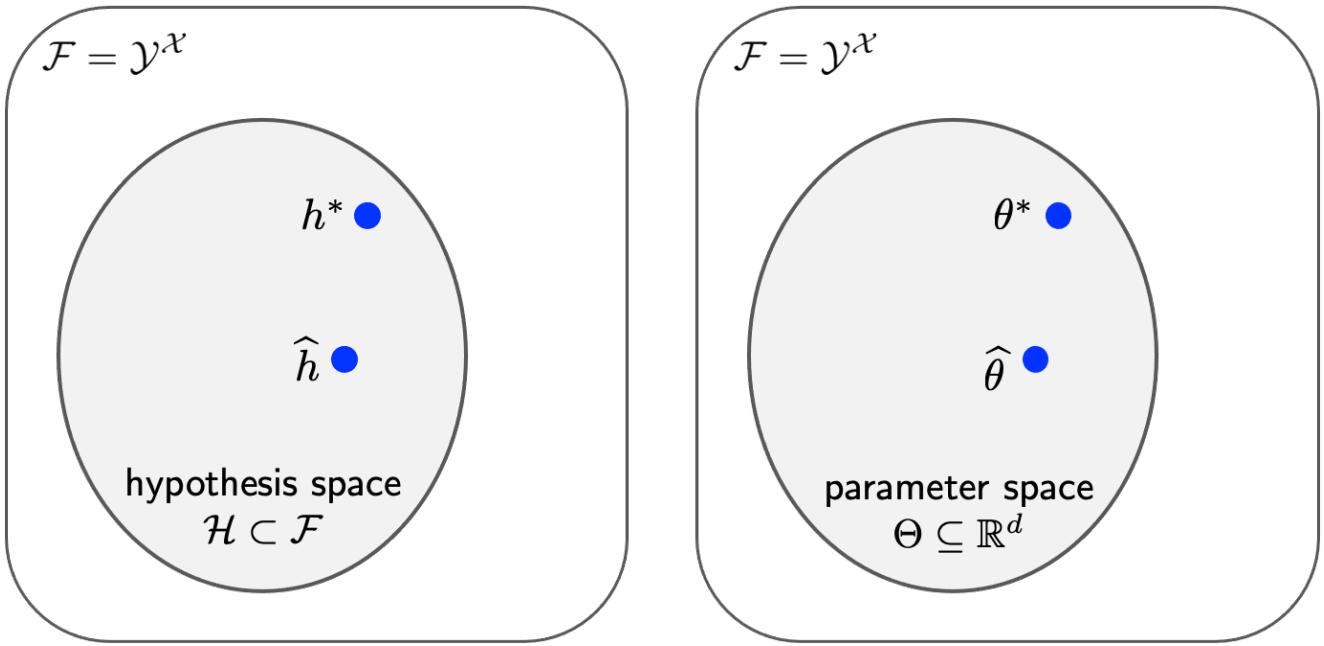
$$p(y \mid \boldsymbol{x}_q) = \int_{h \in \mathcal{H}} p(y \mid \boldsymbol{x}_q, h)\, d\, P(h \mid \mathcal{D})$$

- Bayesian inference is extremely costly but can be done approximately, for example, using **ensemble methods**.
- **MAP** inference with

$$h_{MAP} = \arg\max_{h \in \mathcal{H}} p(h \mid \mathcal{D})\,.$$

Gaussian processes

- The **hypothesis space** in Bayesian inference is often **parametrized** in the sense that each hypothesis $h = h_{\boldsymbol{\theta}} \in \mathcal{H}$ is (uniquely) identified by a parameter (vector) $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d$ of fixed dimensionality d.



Gaussian processes

- The **hypothesis space** in Bayesian inference is often **parametrized** in the sense that each hypothesis $h = h_{\boldsymbol{\theta}} \in \mathcal{H}$ is (uniquely) identified by a parameter (vector) $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d$ of fixed dimensionality d.
- Thus, computation of the **posterior** essentially comes down to updating beliefs about the true (or best) parameter:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) \propto p(\boldsymbol{\theta}) \cdot p(\mathcal{D} \mid \boldsymbol{\theta}).$$

- **Gaussian processes** generalize the Bayesian approach from inference about multivariate (but finite-dimensional) random variables to inference about (infinite-dimensional) functions.
- Thus, they can be thought of as **distributions** not just over random vectors but **over random functions**.

Gaussian processes

- A stochastic process in the form of a collection of random variables

$$\{f(\boldsymbol{x}) \,|\, \boldsymbol{x} \in \mathcal{X}\}$$

with index set $\mathcal{X}$ is said to be drawn from a Gaussian process with **mean function** m and **covariance function** k, denoted $f \sim \mathcal{GP}(m, k)$, if for any finite set of elements

$$\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \mathcal{X}\,,$$

the associated finite set of random variables $f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_m)$ has the following multivariate normal distribution:

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \vdots \\ f(\boldsymbol{x}_m) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{x}_1) \\ m(\boldsymbol{x}_2) \\ \vdots \\ m(\boldsymbol{x}_m) \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_m) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_m, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_m, \boldsymbol{x}_m) \end{bmatrix} \right)$$

Gaussian processes

- The above properties imply

$$m(\boldsymbol{x}) = \mathbb{E}(f(\boldsymbol{x}))\,,$$
$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}\big((f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x}') - m(\boldsymbol{x}'))\big)\,.$$

- Moreover, k needs to obey the properties of a **kernel function** (so as to guarantee proper covariance matrices).

- Intuitively, a function f drawn from a Gaussian process prior can be thought of as a (very) high-dimensional vector drawn from a (very) high-dimensional multivariate Gaussian.

- Here, each dimension of the Gaussian corresponds to an element $\boldsymbol{x}$ from the index set $\mathcal{X}$, and the corresponding component of the random vector represents the value of $f(\boldsymbol{x})$.

Gaussian processes

- Gaussian processes allow for doing proper Bayesian inference in a **non-parametric** way.
- Starting with a **prior** on functions $h \in \mathcal{H}$, specified by a mean function m (often zero) and kernel k, this distribution is replaced by a **posterior** in light of observed data

$$\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N} .$$

- An observation

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i$$

  could be corrupted by an additional (additive) noise component ϵ_i.
- Likewise, a **posterior predictive distribution** can be obtained on outcomes $y \in \mathcal{Y}$ for a new query $\boldsymbol{x}_q \in \mathcal{X}$.

Gaussian processes

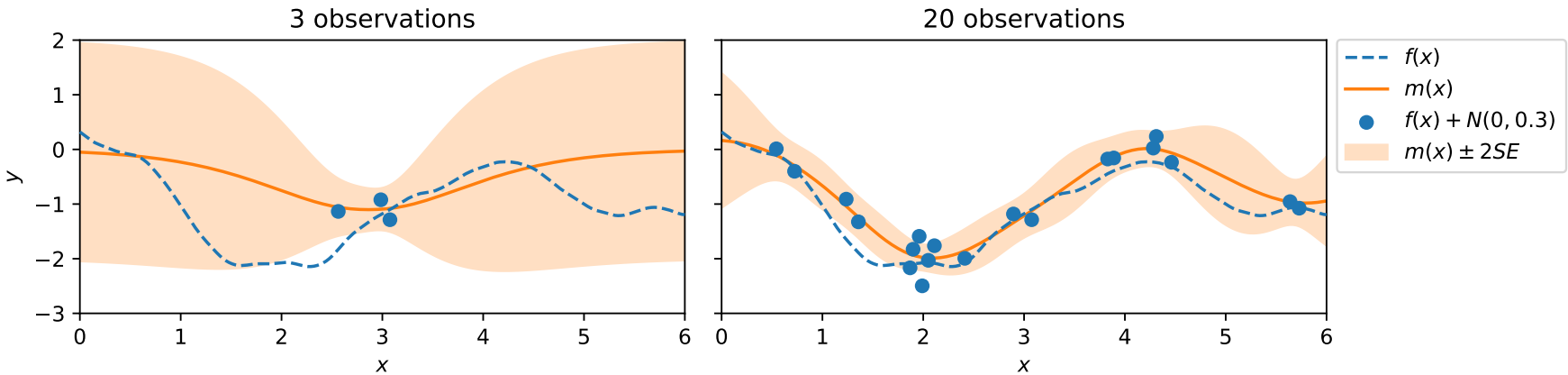
- In general, the computations are intractable, though turn out to be rather simple in the case of regression ($\mathcal{Y} = \mathbb{R}$) with Gaussian noise.

- In this case, and assuming a zero-mean prior, the posterior predictive distribution is again a Gaussian with mean μ and variance σ^2:

$$\mu = K(\boldsymbol{x}_q, X)(K(X, X) + \sigma_\epsilon^2 I)^{-1}\boldsymbol{y}\,,$$
$$\sigma^2 = K(\boldsymbol{x}_q, \boldsymbol{x}_q) + \sigma_\epsilon^2 - K(\boldsymbol{x}_q, X)(K(X, X) + \sigma_\epsilon^2 I)^{-1}K(X, \boldsymbol{x}_q)\,,$$

  where
  - $(X, \boldsymbol{y})$ is the data,
  - $K(X, X)$ is the kernel matrix with entries $(K(X, X))_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$,
  - σ_ϵ^2 is the variance of the (additive) error term for the observations.

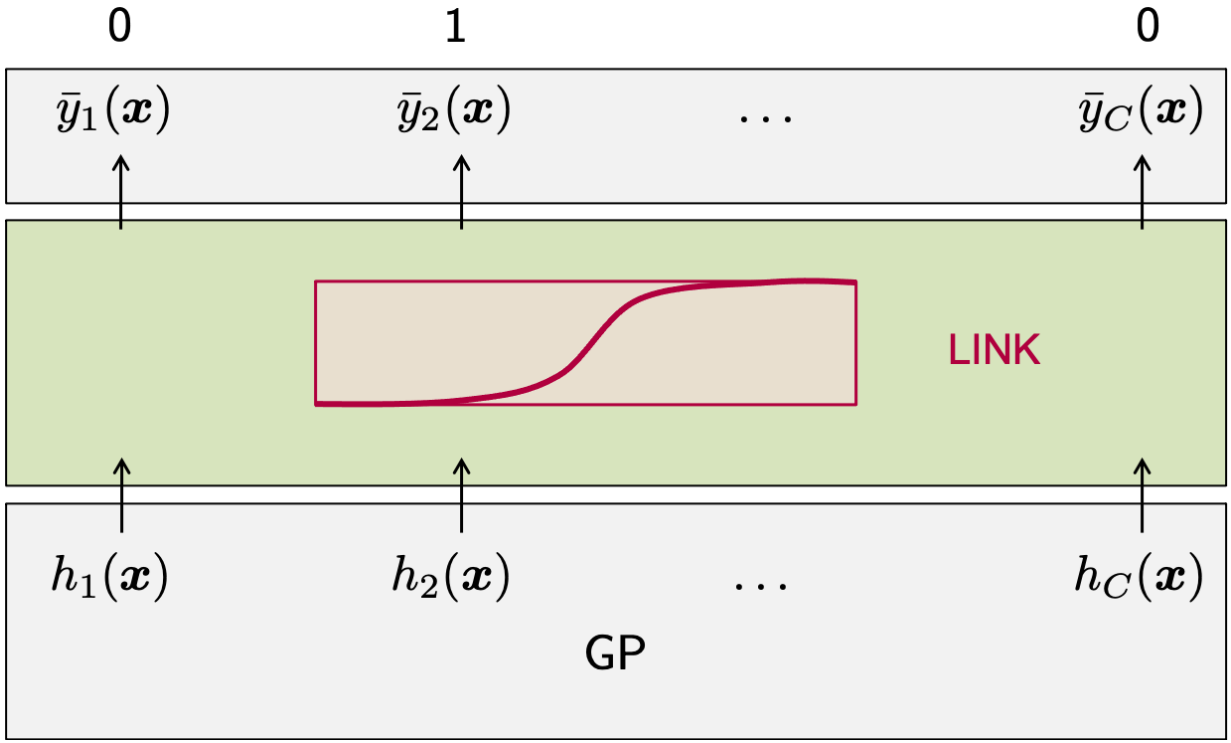
Gaussian processes



Simple one-dimensional illustration of Gaussian processes ($\mathcal{X} = [0, 6]$), with very few examples on the left and more examples on the right. The predictive uncertainty, as reflected by the width of the confidence band around the mean function (orange line) reduces with an increasing number of observations.

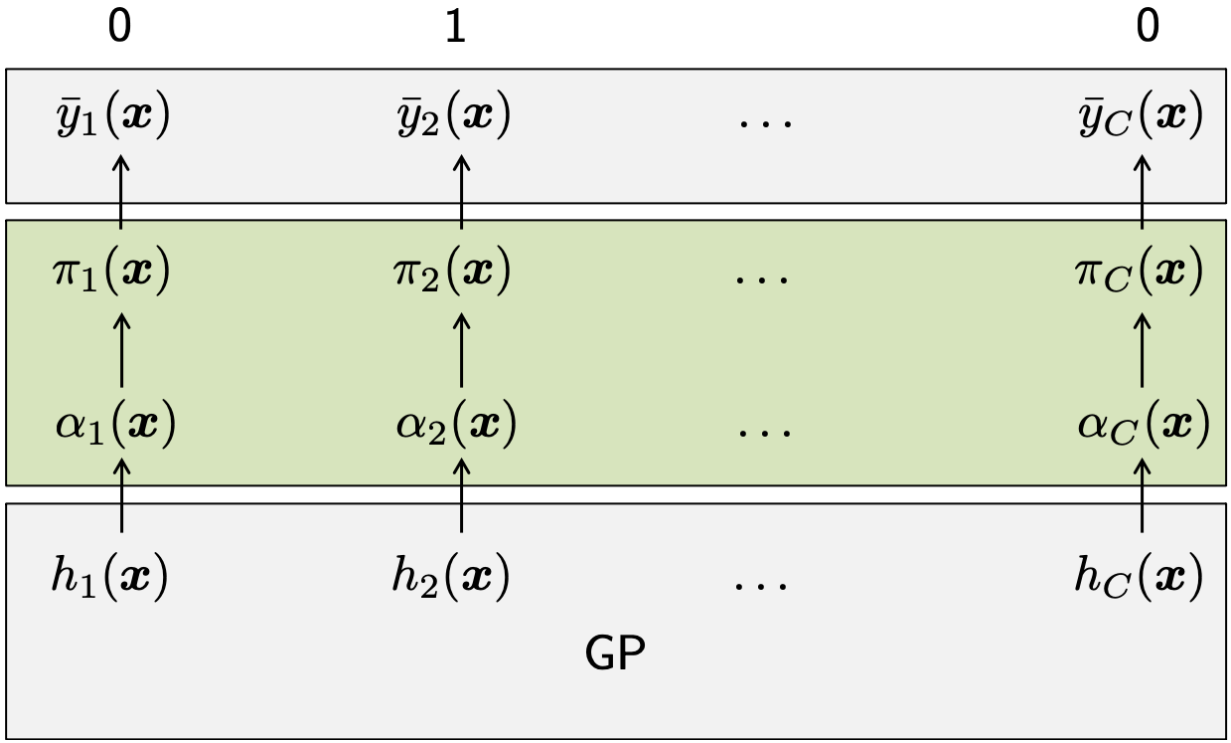
Gaussian processes

- Problems with **discrete outcomes** $y \in \mathcal{Y} = \{\bar{y}_1, \ldots, \bar{y}_C\}$ are made amenable to Gaussian processes by linking these outcomes with the real values $h = h(\boldsymbol{x})$ as underlying (latent) variables.



Gaussian processes

- Problems with **discrete outcomes** $y \in \mathcal{Y} = \{\bar{y}_1, \ldots, \bar{y}_C\}$ are made amenable to Gaussian processes by linking these outcomes with the real values $h = h(\boldsymbol{x})$ as underlying (latent) variables.



Gaussian processes

- Problems with **discrete outcomes** $y \in \mathcal{Y} = \{\bar{y}_1, \ldots, \bar{y}_C\}$ are made amenable to Gaussian processes by linking these outcomes with the real values $h = h(\boldsymbol{x})$ as underlying (latent) variables.
- For example, one can assume that the (conditional) class probabilities are coming from a **Dirichlet distribution**:

$$y \sim \mathrm{Cat}(\pi_1, \ldots, \pi_C), \quad (\pi_1, \ldots, \pi_C) \sim \mathrm{Dir}(\alpha_1, \ldots, \alpha_C),$$

  i.e.,

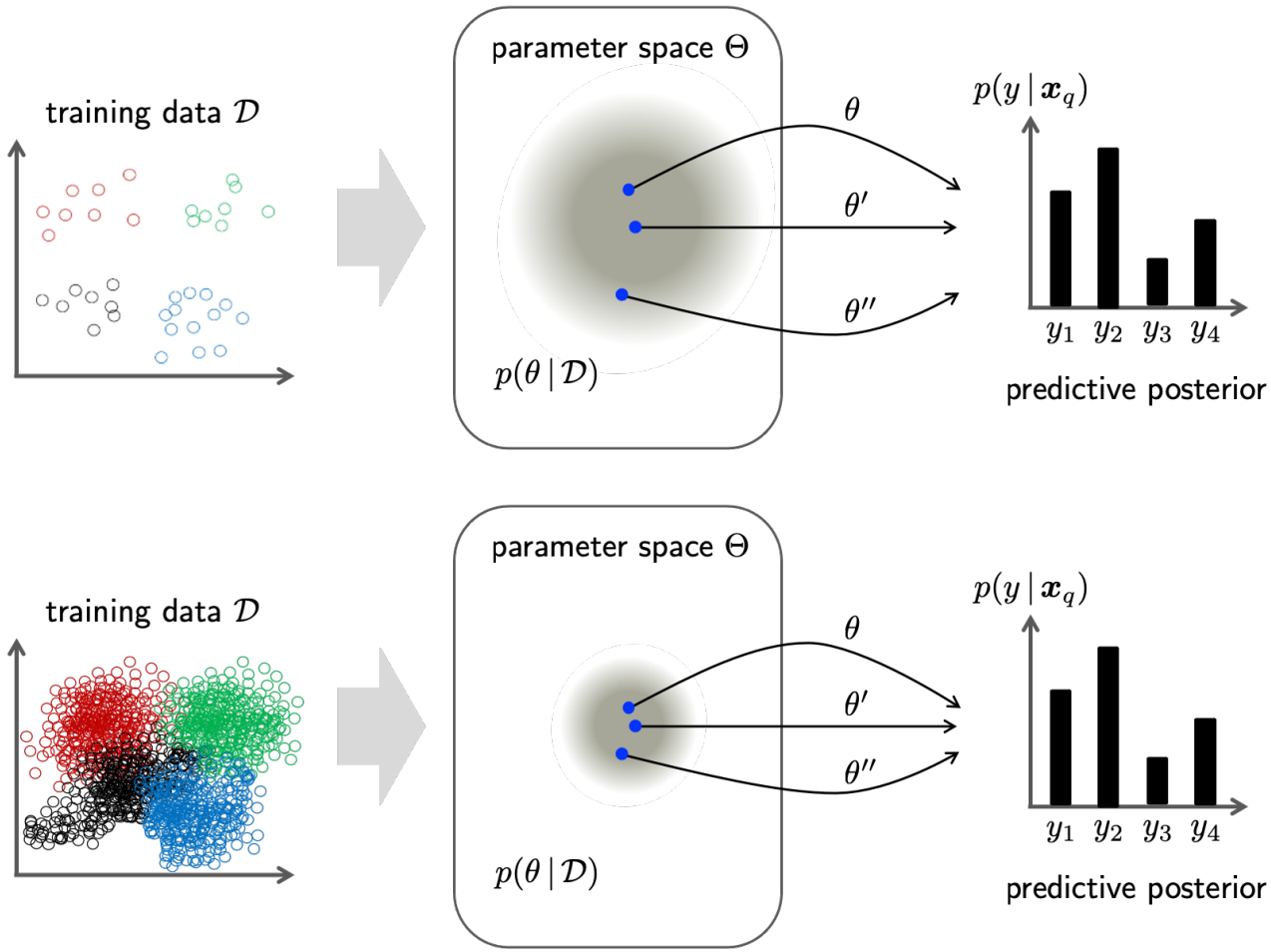
$$p(\pi_1, \ldots, \pi_C) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{C} \pi_i^{\alpha_i - 1}.$$

- The idea, then, is to model the concentration parameters α_i through (latent) GPs f_i.[2]
- Since the likelihood will no longer be Gaussian, **approximate inference** (e.g., Laplace, expectation propagation, MCMC) is needed.

[2] D. Milios, R. Camoriano, P. Michiardi, L. Rosasco, M. Filippone. Dirichlet-based Gaussian Processes for Large-scale Calibrated Classification. NeurIPS 2018.

Bayesian learning

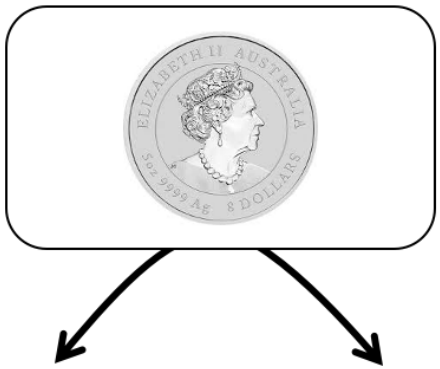
- The posterior $p(h \mid \mathcal{D})$ contains valuable information about the learner's **state of knowledge**:

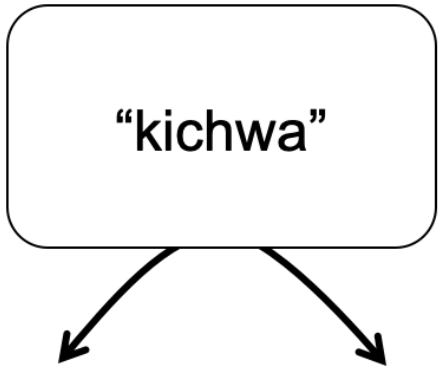


Aleatoric versus epistemic uncertainty

- **Aleatoric** (**aka** statistical) uncertainty refers to the notion of **randomness**, that is, the variability in the outcome of an experiment which is due to inherently random effects.
- **Epistemic** (**aka** systematic) uncertainty refers to uncertainty caused by a **lack of knowledge**, i.e., to the epistemic state of the agent.
- As opposed to aleatoric uncertainty, epistemic uncertainty can in principle be reduced on the basis of additional information.

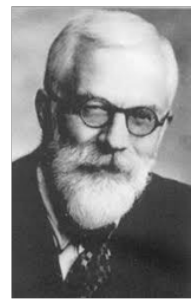
Aleatoric versus epistemic uncertainty





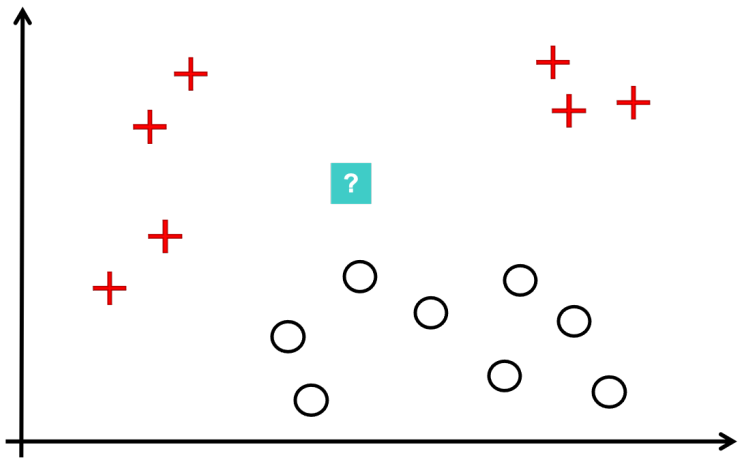
"Not knowing the chance of mutually exclusive events and knowing the chance to be equal are two quite different states of knowledge"

Ronald Fisher (1890-1962)



Aleatoric versus epistemic uncertainty in ML

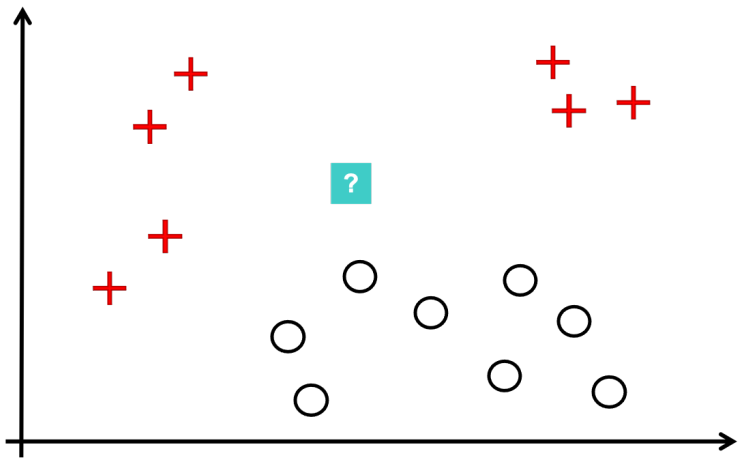
- Both types of uncertainty also play an important role in ML[3], where the learner's state of knowledge strongly depends on the amount of data seen so far ...

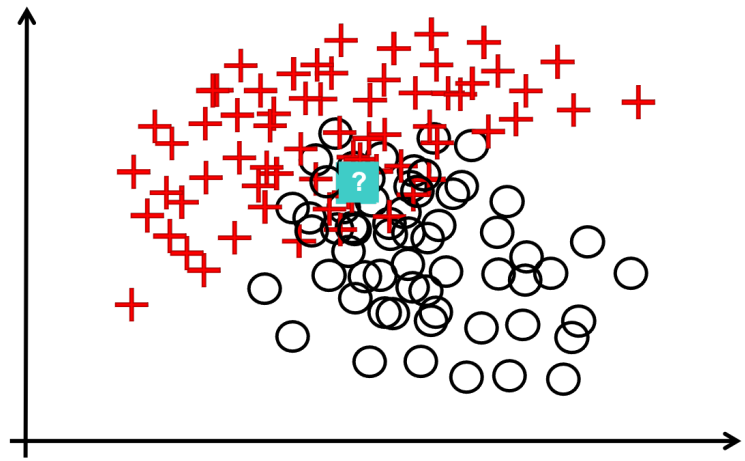


[3] E. Hüllermeier, W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. Machine Learning, 110(3):457-506, 2021.

Aleatoric versus epistemic uncertainty in ML

- Both types of uncertainty also play an important role in ML[3], where the learner's state of knowledge strongly depends on the amount of data seen so far ...

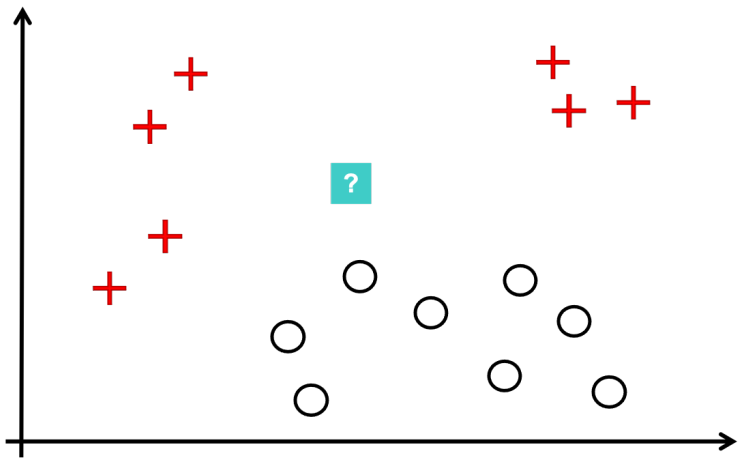


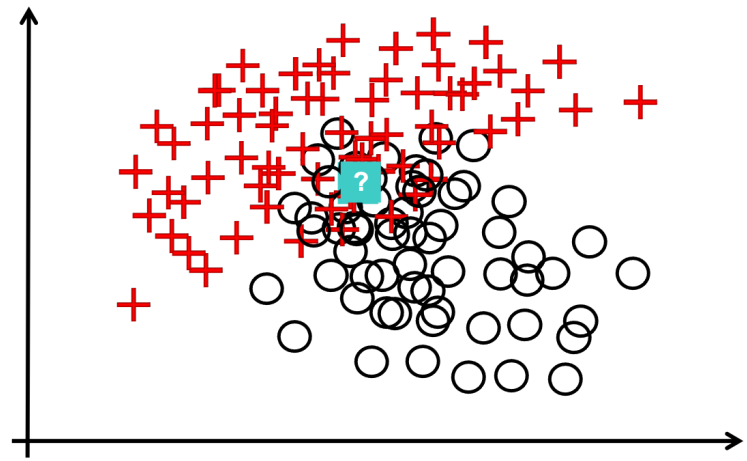


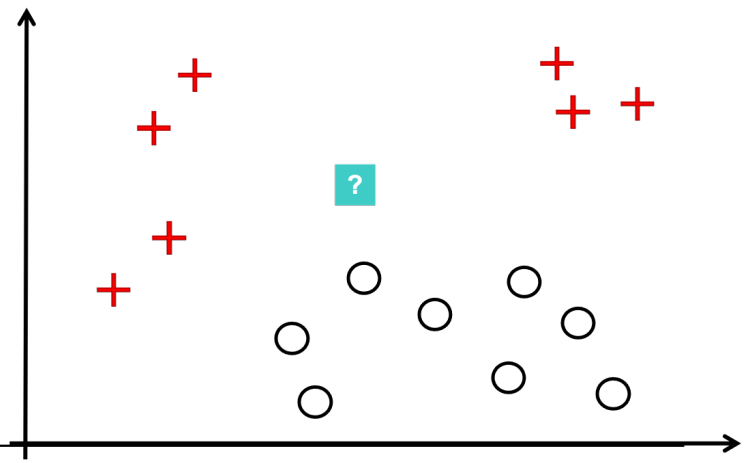
[3] E. Hüllermeier, W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. Machine Learning, 110(3):457-506, 2021.

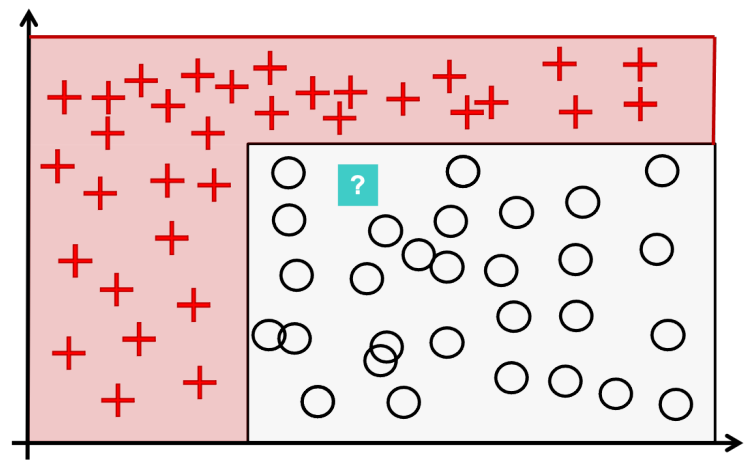
Aleatoric versus epistemic uncertainty in ML

- Both types of uncertainty also play an important role in ML[3], where the learner's state of knowledge strongly depends on the amount of data seen so far ...





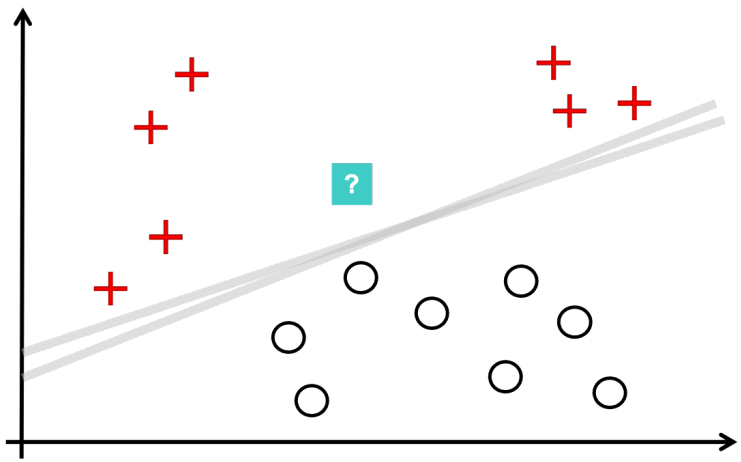




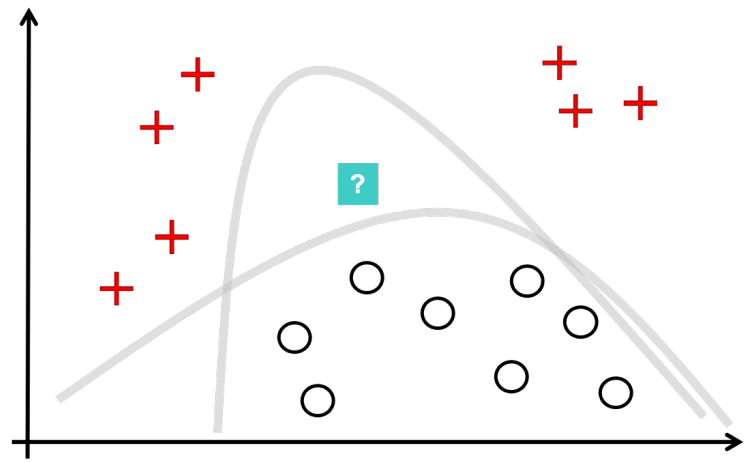
[3] E. Hüllermeier, W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. Machine Learning, 110(3):457-506, 2021.

Aleatoric versus epistemic uncertainty in ML

- ... but also on the underlying model assumptions:



strong prior



weaker prior

Aleatoric versus epistemic uncertainty in ML

- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?

Aleatoric versus epistemic uncertainty in ML

- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?
- Predict the next number: 116, 304, 194, 341, 224, 654, 609, 625, 533, 91, 205, 35, 527, 611, 128, 235, 348, 912, 582, 52, 672, 20, 856, 904, 628, 273, 615, 105, 610, 862, 384, 705, 73, 794, 775, 156, **??**

Aleatoric versus epistemic uncertainty in ML

- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?
- Predict the next number: 116, 304, 194, 341, 224, 654, 609, 625, 533, 91, 205, 35, 527, 611, 128, 235, 348, 912, 582, 52, 672, 20, 856, 904, 628, 273, 615, 105, 610, 862, 384, 705, 73, 794, 775, 156, **??**

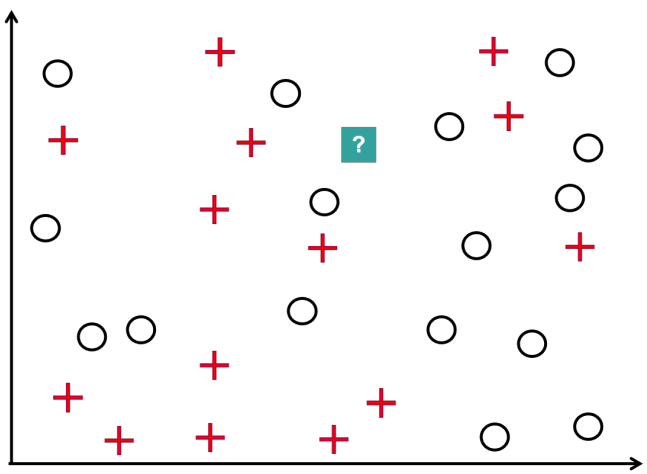
$$x \leftarrow x \times 237 \bmod 971$$

Aleatoric versus epistemic uncertainty in ML

- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?
- Predict the next number: 116, 304, 194, 341, 224, 654, 609, 625, 533, 91, 205, 35, 527, 611, 128, 235, 348, 912, 582, 52, 672, 20, 856, 904, 628, 273, 615, 105, 610, 862, 384, 705, 73, 794, 775, 156, **??**

$$x \leftarrow x \times 237 \bmod 971$$

- Predict the label of the query instance:

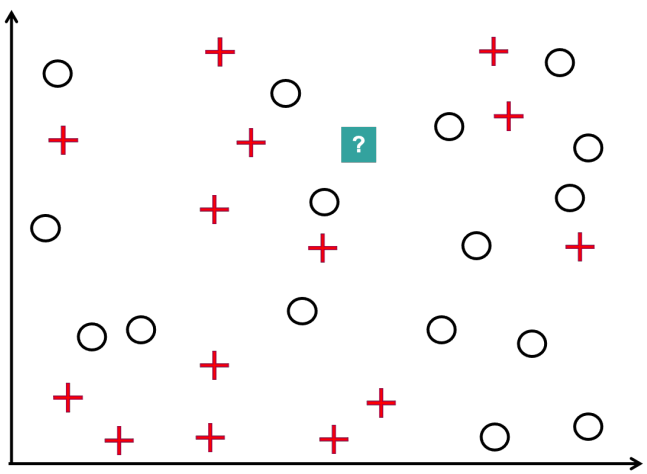


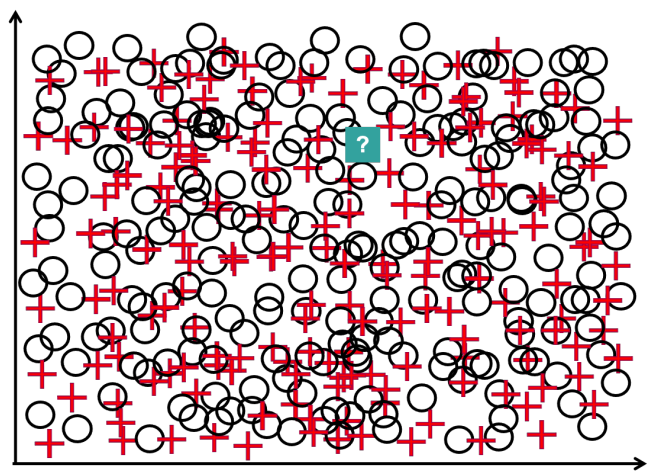
Aleatoric versus epistemic uncertainty in ML

- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?
- Predict the next number: 116, 304, 194, 341, 224, 654, 609, 625, 533, 91, 205, 35, 527, 611, 128, 235, 348, 912, 582, 52, 672, 20, 856, 904, 628, 273, 615, 105, 610, 862, 384, 705, 73, 794, 775, 156, **??**

$$x \leftarrow x \times 237 \bmod 971$$

- Predict the label of the query instance:



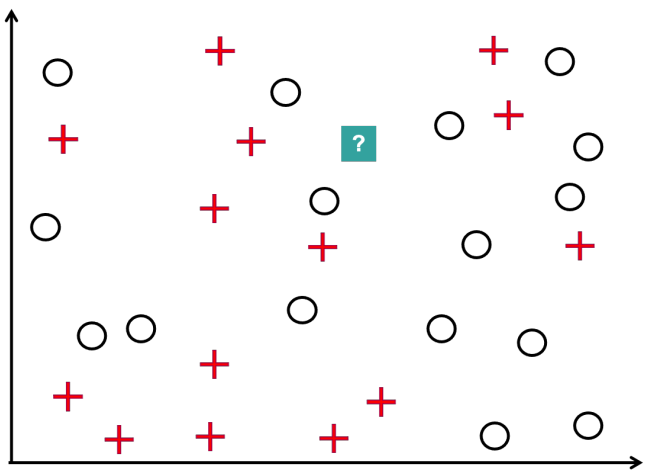


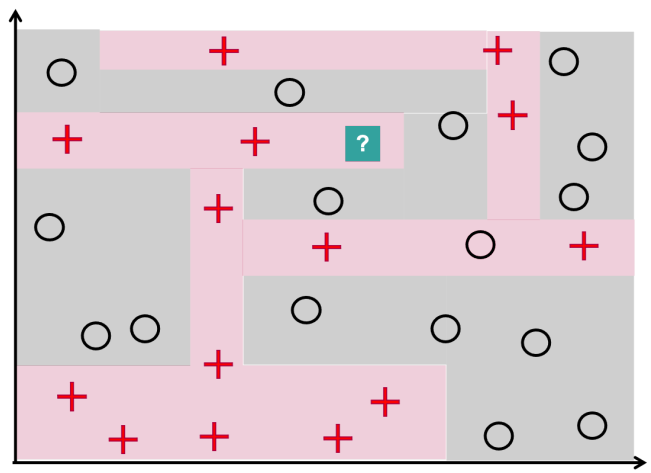
Aleatoric versus epistemic uncertainty in ML

- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?
- Predict the next number: 116, 304, 194, 341, 224, 654, 609, 625, 533, 91, 205, 35, 527, 611, 128, 235, 348, 912, 582, 52, 672, 20, 856, 904, 628, 273, 615, 105, 610, 862, 384, 705, 73, 794, 775, 156, **??**

Aleatoric versus epistemic uncertainty in ML

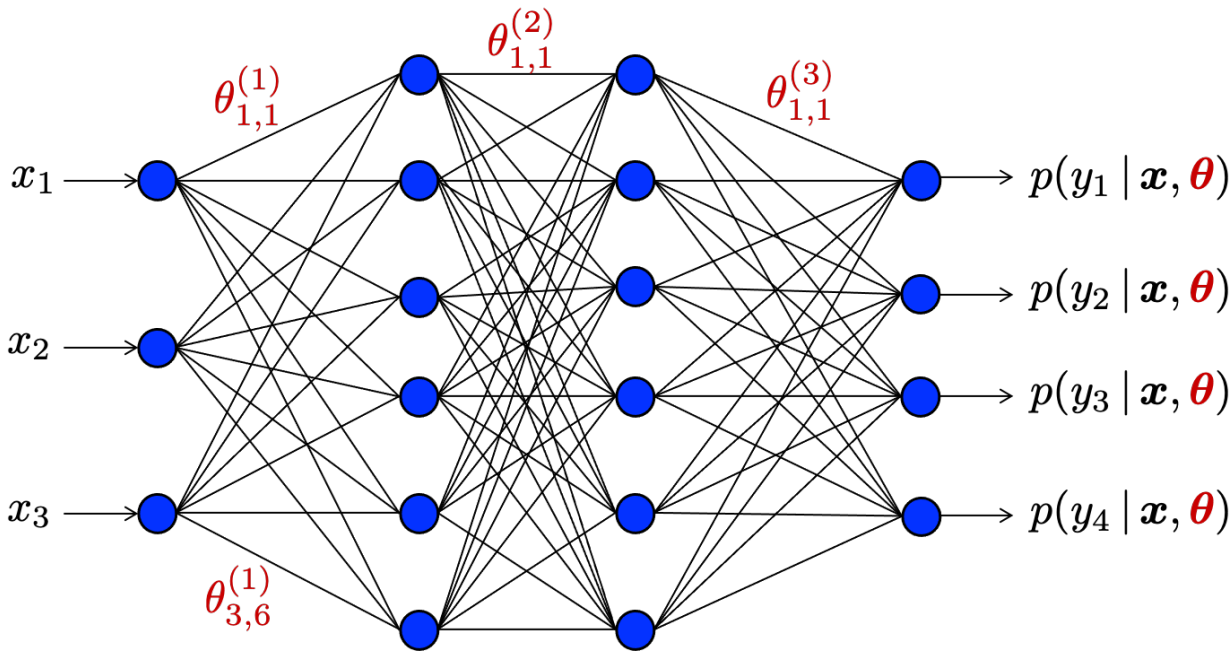
- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?
- Predict the next number: 116, 304, 194, 341, 224, 654, 609, 625, 533, 91, 205, 35, 527, 611, 128, 235, 348, 912, 582, 52, 672, 20, 856, 904, 628, 273, 615, 105, 610, 862, 384, 705, 73, 794, 775, 156, **??**
  $x \leftarrow x \times 237 \bmod 971$
- Predict the label of the query instance:





Uncertainty quantification

- In the case of neural networks, epistemic uncertainty essentially corresponds to uncertainty about the network weights.
- Fixed weights $\boldsymbol{\theta}$ lead to a fixed probability $p(\cdot \mid \boldsymbol{x}, \boldsymbol{\theta})$.



Uncertainty quantification

- How to measure uncertainty, i.e., quantify the amount of uncertainty contained in a prediction?
- A well-known uncertainty measure is the **Shannon entropy**, which, in the case of discrete probability $p : \mathcal{Y} \to [0, 1]$, is given by

$$H\big[\, Y \,\big] = H\big[\, p \,\big] = -\sum_{y \in \mathcal{Y}} p(y) \log_2 p(y) \,.$$

- What we seek is a **decomposition**

$$\underbrace{\mathrm{TU}(\boldsymbol{x})}_{\text{total uncertainty}} = \underbrace{\mathrm{AU}(\boldsymbol{x})}_{\text{aleatoric uncertainty}} + \underbrace{\mathrm{EU}(\boldsymbol{x})}_{\text{epistemic uncertainty}}$$

Uncertainty quantification

- One idea is to model **epistemic uncertainty** as **mutual information** between outcomes and hypotheses:[4]

$$\underbrace{H[Y]}_{\text{total uncertainty}} = \underbrace{I(Y;\Theta)}_{\text{epistemic}} + \underbrace{H[Y \mid \Theta]}_{\text{aleatoric}}$$

- Intuitively, epistemic uncertainty thus captures the amount of information about the model parameters $\boldsymbol{\theta}$ that would be gained through knowledge of the true outcome y.

- **Total uncertainty** = entropy of the predictive posterior distribution, in the case of discrete $\mathcal{Y}$ given by

$$\mathrm{TU}(\boldsymbol{x}) = H\big[\, p(y \mid \boldsymbol{x}) \,\big] = -\sum_{y \in \mathcal{Y}} p(y \mid \boldsymbol{x}) \log_2 p(y \mid \boldsymbol{x}) \,.$$

[4] S. Depeweg, J.M. Hernandez-Lobato, F. Doshi-Velez, S. Udluft. Decomposition of Uncertainty in Bayesian Deep Learning for Efficient and Risk-sensitive Learning. ICML 2018.

Uncertainty quantification

- This uncertainty also includes the (epistemic) uncertainty about the network weights $\boldsymbol{\theta}$, but fixing a set of weights, i.e., considering a distribution $p(y \mid \boldsymbol{x}, \boldsymbol{\theta})$, removes the epistemic uncertainty.
- Therefore, the expectation over the entropies of these distributions,

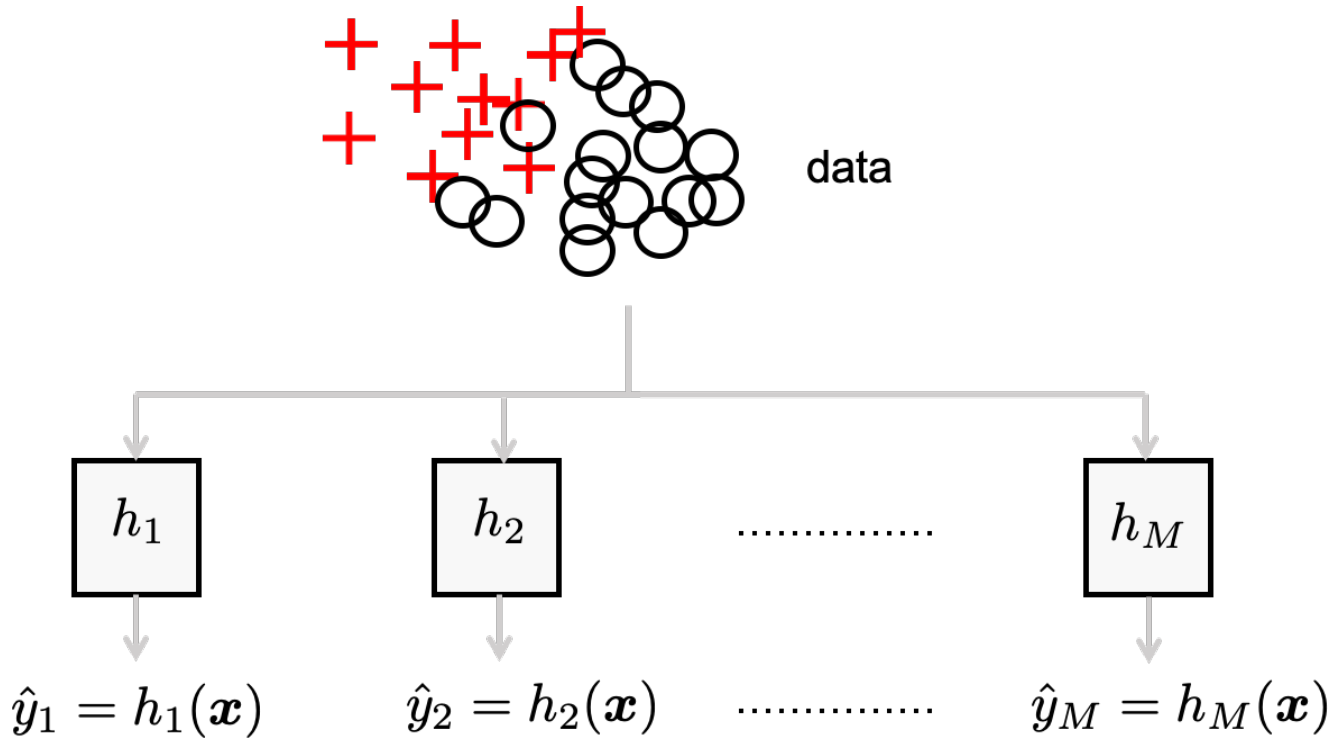
$$\mathbb{E}_{p(\boldsymbol{\theta} \mid \mathcal{D})} H\big[p(y \mid \boldsymbol{x}, \boldsymbol{\theta})\big] = \\ = -\int p(\boldsymbol{\theta} \mid \mathcal{D}) \left(\sum_{y \in \mathcal{Y}} p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) \log_2 p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) \right) d\,\boldsymbol{\theta} \ ,$$

  is a measure of the **aleatoric uncertainty** (conditional entropy).
- Finally, the **epistemic uncertainty** is obtained as the difference

$$\mathrm{EU}(\boldsymbol{x}) := H\big[p(y \mid \boldsymbol{x})\big] - \mathbb{E}_{p(\boldsymbol{\theta} \mid \mathcal{D})} H\big[p(y \mid \boldsymbol{x}, \boldsymbol{\theta})\big] \ ,$$

  which equals the **mutual information** between y and $\boldsymbol{\theta}$.

Ensemble methods for uncertainty quantification



- Ensemble can be seen as an approximation of a distribution.
- Intuitively, diversity is an indicator of epistemic uncertainty.

Bayesian agents: Ensemble-based approximation

- Based on an ensemble of hypotheses $h_1, \ldots, h_M$, producing respective predictions $p_1, \ldots, p_M$, an approximation of conditional entropy can be obtained by

$$\mathrm{AU}(\boldsymbol{x}) := -\frac{1}{M} \sum_{i=1}^{M} \sum_{y \in \mathcal{Y}} p_i(y \mid \boldsymbol{x}) \log_2 p_i(y \mid \boldsymbol{x}) \,,$$

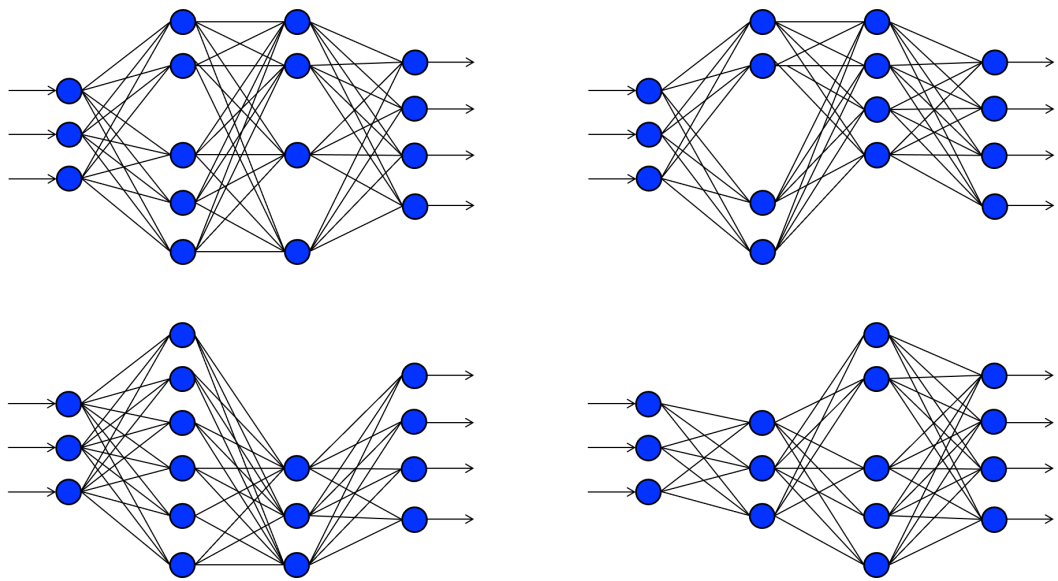
  an approximation of total uncertainty (Shannon entropy) by

$$\mathrm{TU}(\boldsymbol{x}) := -\sum_{y \in \mathcal{Y}} \underbrace{\left(\frac{1}{M} \sum_{i=1}^{M} p_i(y \mid \boldsymbol{x}) \right)}_{\bar{p}(y \mid \boldsymbol{x})} \log_2 \underbrace{\left(\frac{1}{M} \sum_{i=1}^{M} p_i(y \mid \boldsymbol{x}) \right)}_{\bar{p}(y \mid \boldsymbol{x})} ,$$

  and an approximation of epistemic uncertainty (mutual information) by the difference, which is equivalent to **Jensen-Shannon divergence** of the distributions $p_i(y \mid \boldsymbol{x})$, $i = 1, \ldots, M$.

Bayesian agents: Ensemble-based approximation

- For neural networks, it has been shown that techniques such as **Dropout**[5] and **DropConnect**[6] can be interpreted as (implicit) ensemble methods, and can hence be used to implement this approach.
- Of course, any other ensemble technique could be used as well.



[5] Y. Gal, Z. Ghahramani. Bayesian convolutional neural networks with Bernoulli approximate variational inference. ICLR Workshop Track, 2016.

[6] A. Mobiny, H.V. Nguyen, S. Moulik, N. Garg, C.C. Wu. DropConnect is effective in modeling uncertainty of Bayesian networks. Scientific Reports, 11(5458), 2021.

Summary and Outlook

- **Uncertainty** is of major importance in ML in general.
- We also highlighted the benefits of distinguishing between different types of uncertainty, notably **aleatoric and epistemic uncertainty**.
- Currently, new proposals for modeling and quantifying uncertainty appear on a regular basis.
- Eventually, it would be desirable to "derive" a measure of total uncertainty as well as its decomposition into aleatoric and epistemic parts on an **axiomatic basis**.
- How to perform an **empirical evaluation** of methods for quantifying uncertainty? ($\rightarrow$ rejection)
- Most approaches so far neglect **model uncertainty**, assuming instead that the model is correctly specified, altough **model misspecification** is a common problem in practice.
- Related to this is the **"closed world" assumption**, which is often violated in practice, e.g., in the case of OOD data.

Overview of this talk

Overview of this talk

1. Introduction
2. Uncertainty in classification: representation and methods (40 minutes)
3. Coffee break
4. Classification with reject option (25 minutes)
5. Set-valued prediction methods (25 minutes)
6. Coffee break
7. Rejection and set-valued prediction with distribution shift (40 minutes)

Leveraging uncertainty information

Now that we have seen the different forms of uncertainties and ways to represent and estimate them, one question remains...

What should we do with this uncertainty information?

$\Rightarrow$ Adapt our decision-making process

Making decisions in presence of uncertainty

Standard classification

$h : \quad \mathcal{X} \quad \rightarrow \quad \mathcal{Y}$

$\mapsto$ cookie

Classification with reject option

$(h, r) : \quad \mathcal{X} \quad \rightarrow \quad \mathcal{Y} \cup \{\text{“I don't know”}\}$

$\mapsto$ “I don't know”

Decision-theoretic perspective to rejection

We modify the standard classifier

$$h : \mathcal{X} \to \mathcal{Y}$$

by adding a **rejector** $r : \mathcal{X} \to \{0, 1\}$ to it such that

$$(h, r)(x) = \begin{cases} h(x) & \text{if } r(x) = 0, \\ \emptyset & \text{otherwise (r(x) = 1)} \end{cases}$$

What properties should the rejector r satisfy?

Decision-theoretic perspective to rejection: Chow's rule[8]

Assume that rejecting induces a fixed cost $\lambda > 0$, we can define the **associated risk** of misclassification as[7]

$$R_\lambda(h, r) := \mathbb{E}_{\boldsymbol{X},Y}\left[\mathbb{1}_{Y \neq h(\boldsymbol{X})}\mathbb{1}_{r(\boldsymbol{X})=0} + \lambda\,\mathbb{1}_{r(\boldsymbol{X})=1}\right].$$

The **Bayes optimal predictor** of this risk is a thresholding criterion:

$$h^*(\boldsymbol{x}) = \arg\max_k p(y = k \mid \boldsymbol{x}),$$

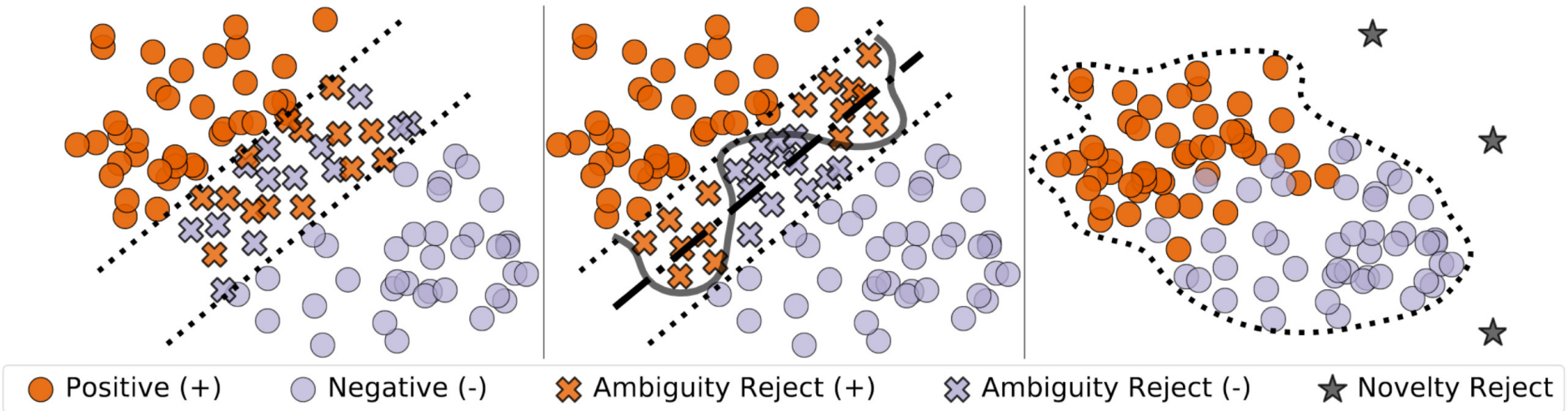
$$r^*(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \max_k p(y = k \mid \boldsymbol{x}) < 1 - \lambda, \\ 0 & \text{otherwise.} \end{cases}$$

$\Rightarrow$ **Reject ambiguous samples**

[7] Herbei et al., Classification with reject option, 2006

[8] Chow, An optimum character recognition system using decision functions, 1957

The different types of rejection



Picture taken from: Hendrickx et al., Machine Learning with a Reject Option: A survey, Arxiv 2021.

Properties of the classifier-rejector relationship[11]

The architecture of the classifier can be partly shared with the rejector: it can be totally **separated**, **dependent** or **integrated**.

The training of the rejector can be done either **sequentially** or **simultaneously**.

Examples of simultaneous learning:

- hinge loss similar to SVM[9]
- SelectiveNet neural network[10]

[9] Bartlett et al., Classification with a Reject Option using a Hinge Loss, JMLR 2008.

[10] Geifman et al., SelectiveNet: A Deep Neural Network with an Integrated Reject Option, ICML 2019.

[11] Hendrickx et al., Machine Learning with a Reject Option: A survey, Arxiv 2021.

Overview of this talk

1. Introduction
2. Uncertainty in classification: representation and methods (40 minutes)
3. Coffee break
4. Classification with reject option (25 minutes)
5. Set-valued prediction methods (25 minutes)
6. Coffee break
7. Rejection and set-valued prediction with distribution shift (40 minutes)

Predicting sets rather than single labels

Standard classification

$h : \quad \mathcal{X} \quad \to \quad \mathcal{Y}$

$\mapsto$ cookie

Classification with reject option

$(h, r) : \quad \mathcal{X} \quad \to \quad \mathcal{Y} \cup \{\text{“I don't know”}\}$

$\mapsto$ “I don't know”

Set-valued classification

$\Gamma : \quad \mathcal{X} \quad \to \quad \mathcal{P}(\mathcal{Y})$

$\mapsto \{\text{cookie}, \text{panda}\}$

Why predicting sets?

- single object with uncertainty

epistemic

keyboard

aleatoric

impala, gazelle, hartebeest

Why predicting sets?

- several objects present in the data point but only a single one was labeled

banana, orange,
Granny Smith

sports car, convertible

"The scale of ILSVRC classification task [...] makes it very expensive to label every instance of every object in every image. Therefore, on this dataset only one object category is labeled in each image. This creates ambiguity in evaluation."[12]

[12] Russakovsky et al., ImageNet Large Scale Visual Recognition Challenge, 2015.

Differences and similarities with multi-label classification

- Differences
  - we are only given **a single label per sample** during both training and evaluation
  - this induces a much **lower annotation cost**
- Connections
  - partial multi-label classification:[13] a subset of the full label sets are known during training;
  - single positive labels classification:[14] only a single positive label is known during training, but, for evaluation, the full label sets are known.

[13] Durand et al., Learning a deep convnet for multi-label classification with partial labels, CVPR 2019.

[14] Cole et al., Multi-label learning from single positive labels, CVPR 2021.

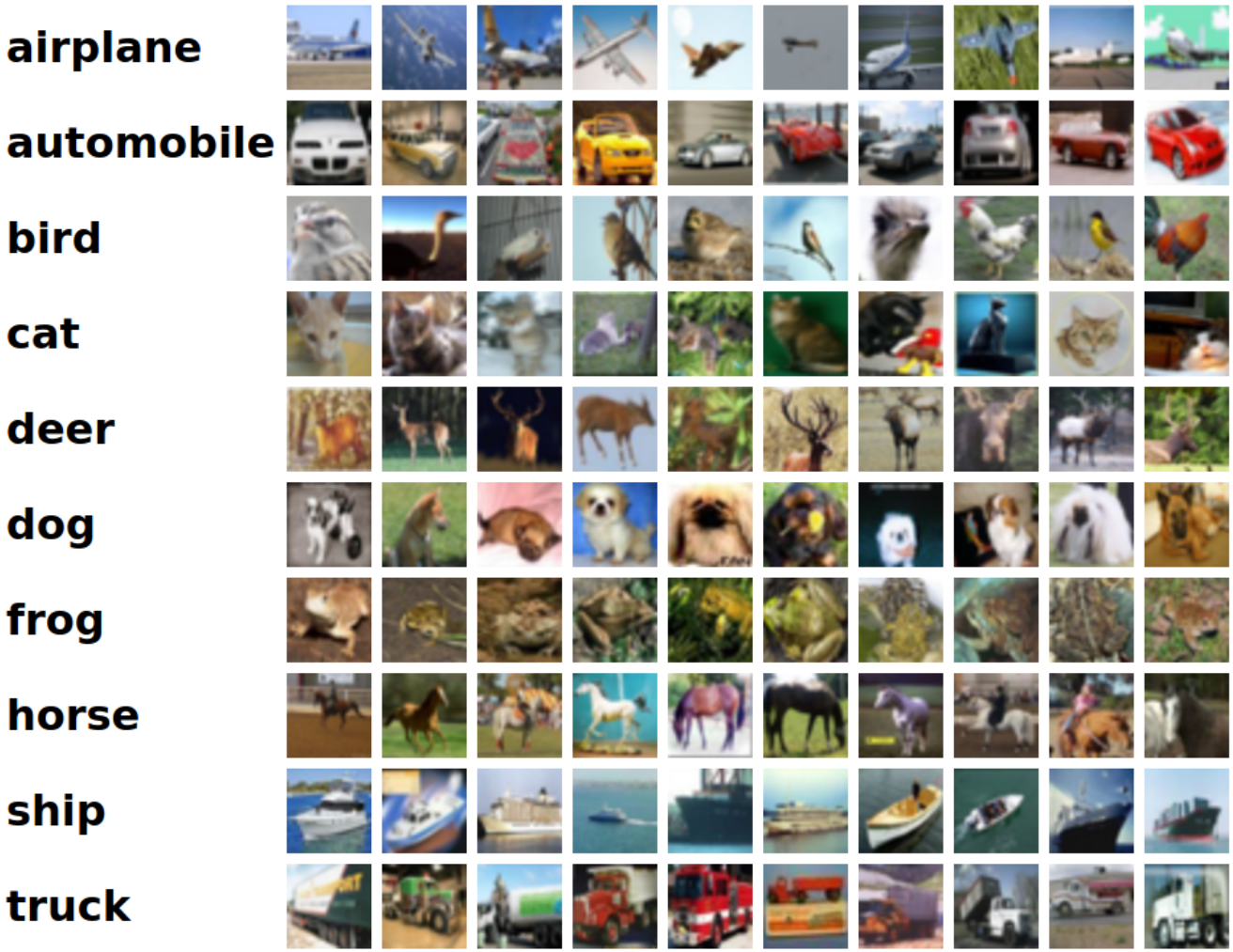
How to predict sets?

- Easier approach: **top-K prediction**
  - predict the K most probable classes
  - common approach: top-5 error rate used as ImageNet's official metric
- Efficient:
  - can be easily applied after fitting a model
  - explicit optimization of the top-K error rate possible
    - ★ top-K hinge losses[15]
    - ★ smooth losses adapted for neural network training[16]
- Issue: **not adaptive**
  - always the same set size predicted independently of the sample.

[15] Lapin et al., Loss functions for top-k error: Analysis and insights, CVPR 2018.

[16] Berrada et al., Smooth loss functions for deep top-k classification, ICLR 2018.

Human uncertainty in CIFAR-10 dataset[17]



[17] Krizhevsky et al., Learning multiple layers of features from tiny images, 2009.

Human uncertainty in CIFAR-10 dataset

CIFAR-10H[18]: reannotation of CIFAR-10 test set, keeping all the human annotations (top-1 error rate: 4.56%)

[18] Peterson et al., Human uncertainty makes classification more robust, ICCV 2016.

How to predict sets?

Trade-off between two quantities:

- correctness: quantified using **set error rate**
- informativeness: linked to **set size**

	Point-wise	Average
Error rate $\mathcal{E}(\Gamma)$	$\mathbb{P}\left(Y \notin \Gamma(\boldsymbol{X}) \mid \boldsymbol{X}\right)$	$\mathbb{P}\left(Y \notin \Gamma(\boldsymbol{X})\right)$
Set size $\mathcal{I}(\Gamma)$	$\lvert\Gamma(\boldsymbol{x})\rvert$	$\mathbb{E}\left[\lvert\Gamma(\boldsymbol{X})\rvert\right]$

How to set this trade-off?

A lot of possible ways to set the trade-off

- linear averaging / penalized risk:[19] $R_\lambda(\Gamma) := \mathcal{E}(\Gamma) + \lambda\,\mathcal{I}(\Gamma)$
- optimizing one quantity under a constraint on the other one:[20]

$$\min_{\Gamma} \ \mathcal{F}(\Gamma) \quad \text{s.t.} \quad \Gamma \in \mathcal{C}$$

- harmonic averaging / F-measure: nondeterministic classifiers[21]

$$F_\beta(\Gamma) := \mathbb{E}_{\boldsymbol{X},Y}\left[\frac{(1+\beta^2)\mathbb{1}_{Y\in\Gamma(\boldsymbol{X})}}{\beta^2 + |\Gamma(\boldsymbol{X})|}\right]$$

- utility maximization methods[22]: $U(\Gamma) := \mathbb{E}_{\boldsymbol{X},Y}\left[g(|\Gamma(\boldsymbol{X})|)\,\mathbb{1}_{Y\in\Gamma(\boldsymbol{X})}\right]$, where $g : \mathbb{N} \to [0,1]$ is a function penalizing large sets.
- ...

[19] Ha, An optimum class-selective rejection rule for pattern recognition, 1996.

[20] Chzhen et al., Set-valued classification–overview via a unified framework, Arxiv 2021

[21] Coz et al., Learning nondeterministic classifiers, JMLR 2009.

[22] Mortier et al., Efficient set-valued prediction in multi-class classification, 2021

A lot of possible ways to set the trade-off

- linear averaging / penalized risk:[19] $R_\lambda(\Gamma) := \mathcal{E}(\Gamma) + \lambda \mathcal{I}(\Gamma)$
- optimizing one quantity under a constraint on the other one:[20]

$$\min_{\Gamma} \mathcal{F}(\Gamma) \quad \text{s.t.} \quad \Gamma \in \mathcal{C}$$

- harmonic averaging / F-measure: nondeterministic classifiers[21]

$$F_\beta(\Gamma) := \mathbb{E}_{\boldsymbol{X},Y}\left[\frac{(1+\beta^2)\mathbb{1}_{Y\in\Gamma(\boldsymbol{X})}}{\beta^2 + |\Gamma(\boldsymbol{X})|}\right]$$

- utility maximization methods[22]: $U(\Gamma) := \mathbb{E}_{\boldsymbol{X},Y}\left[g(|\Gamma(\boldsymbol{X})|)\,\mathbb{1}_{Y\in\Gamma(\boldsymbol{X})}\right]$, where $g : \mathbb{N} \to [0,1]$ is a function penalizing large sets.
- ...

[19] Ha, An optimum class-selective rejection rule for pattern recognition, 1996.

[20] Chzhen et al., Set-valued classification–overview via a unified framework, Arxiv 2021

[21] Coz et al., Learning nondeterministic classifiers, JMLR 2009.

[22] Mortier et al., Efficient set-valued prediction in multi-class classification, 2021

Constrained optimization

Minimization under constraint formulation:

$$\Gamma^*_{\mathcal{F},\mathcal{C}} \in \underset{\Gamma:\mathcal{X}\to\mathcal{P}(\mathcal{Y})}{\arg\min} \quad \mathcal{F}(\Gamma)$$
$$\text{s.t.} \quad \Gamma \in \mathcal{C}$$

Name	Objective $\mathcal{F}(\Gamma)$	Constraint $\mathcal{C}$	Optimal $\Gamma^*_{\mathcal{F},\mathcal{C}}$
Penalized	$\mathcal{E}(\Gamma) + \lambda\mathcal{I}(\Gamma)$	N/A	Γ^*_λ: threshold
Point-wise size	$\mathcal{E}(\Gamma)$	$\mathcal{I}(\Gamma;\boldsymbol{x}) \leq K$	Γ^*_K: top-K
Average size	$\mathcal{E}(\Gamma)$	$\mathcal{I}(\Gamma) \leq \bar{K}$	$\Gamma^*_{\bar{K}}$: threshold
Point-wise error	$\mathcal{I}(\Gamma)$	$\mathcal{E}(\Gamma;\boldsymbol{x}) \leq \epsilon$	Γ^*_ϵ: top-$K(\boldsymbol{x})$
Average error	$\mathcal{I}(\Gamma)$	$\mathcal{E}(\Gamma) \leq \bar{\epsilon}$	$\Gamma^*_{\bar{\epsilon}}$: threshold

Constrained optimization

Minimization under constraint formulation:

$$\Gamma^*_{\mathcal{F},\mathcal{C}} \in \underset{\Gamma:\mathcal{X}\to\mathcal{P}(\mathcal{Y})}{\arg\min} \quad \mathcal{F}(\Gamma)$$
$$\text{s.t.} \quad \Gamma \in \mathcal{C}$$

Name	Objective $\mathcal{F}(\Gamma)$	Constraint $\mathcal{C}$	Optimal $\Gamma^*_{\mathcal{F},\mathcal{C}}$
Penalized	$\mathcal{E}(\Gamma) + \lambda\mathcal{I}(\Gamma)$	N/A	Γ^*_λ: threshold
Point-wise size	$\mathcal{E}(\Gamma)$	$\mathcal{I}(\Gamma;\boldsymbol{x}) \leq K$	Γ^*_K: top-K
Average size	$\mathcal{E}(\Gamma)$	$\mathcal{I}(\Gamma) \leq \bar{K}$	$\Gamma^*_{\bar{K}}$: threshold
Point-wise error	$\mathcal{I}(\Gamma)$	$\mathcal{E}(\Gamma;\boldsymbol{x}) \leq \epsilon$	Γ^*_ϵ: top-$K(\boldsymbol{x})$
Average error	$\mathcal{I}(\Gamma)$	$\mathcal{E}(\Gamma) \leq \bar{\epsilon}$	$\Gamma^*_{\bar{\epsilon}}$: threshold

Constrained optimization: thresholding strategies

The penalized risk's optimal predictor is easy to compute:

$$R_\lambda(\Gamma) := \mathcal{E}(\Gamma) + \lambda\,\mathcal{I}(\Gamma) = \mathbb{E}_{\boldsymbol{X}}\left[1 + \sum_{\bar{y}\in\Gamma(\boldsymbol{X})} (\lambda - p(y = \bar{y} \mid \boldsymbol{X}))\right].$$

The optimal Bayes predictor Γ^*_λ is thus equal to

$$\Gamma^*_\lambda(\boldsymbol{x}) = \{\bar{y} \in \mathcal{Y} \mid p(y = \bar{y} \mid \boldsymbol{x}) \geq \lambda\}.$$

Both framework with a constraint on the average set size[23] and on the average error rate[24] are optimized by thresholding the class conditional probability $p(y|\boldsymbol{x})$. The threshold is a well-chosen **quantile** value.

[23] Denis et al., Confidence sets with expected sizes for multiclass classification, JMLR 2017

[24] Sadinle et al., Least ambiguous set-valued classifiers with bounded error levels, JASA 2019

Constrained optimization: point-wise error control

Optimization problem:

$$\Gamma_\epsilon^* \in \underset{\Gamma:\mathcal{X}\to\mathcal{P}(\mathcal{Y})}{\arg\min} \quad \mathcal{F}(\Gamma)$$
$$\text{s.t.} \quad \mathbb{P}\left(Y \notin \Gamma(\boldsymbol{X}) \mid \boldsymbol{X}\right) \leq \epsilon$$

The optimal Bayes predictor Γ_ϵ^* consists in predicting the minimum of

classes with higher probability until the sum of their probabilities is greater than $1 - \epsilon$.

Conformal prediction[25]

We still want to learn a set predictor $\Gamma : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ but in a slightly different setting.

Coverage

$$\mathbb{P}\left(Y_{N+1} \in \Gamma(\boldsymbol{X}_{N+1})\right)$$

where the probability $\mathbb{P}$ is taken over $(X_1, Y_1), (X_2, Y_2), \ldots, (X_{N+1}, Y_{N+1})$, i.e., over the training data $\mathcal{D}$ and over the test point (X_{N+1}, Y_{N+1}).

Aim

Provide a **guarantee on coverage** given a miscoverage level $\alpha \in (0, 1)$:

$$\mathbb{P}\left(Y_{N+1} \in \Gamma(\boldsymbol{X}_{N+1})\right) \geq 1 - \alpha.$$

[25] Vovk et al., Algorithmic learning in a random world, 2005.

Full conformal prediction

Conformal predictors are based on **nonconformity measures**:

$$A : (\mathcal{X} \times \mathcal{Y})^{N+1} \to \mathbb{R}$$

and associated **nonconformity scores**:

$$s : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$$

using a thresholding method, for a threshold t,

$$\Gamma(\boldsymbol{x}) = \{y \in \mathcal{Y} \mid s(\boldsymbol{x}, y) \leq t\}$$

Setting the right threshold to achieve the wanted coverage level.

Full conformal prediction: Limitations

Computationally demanding:

- we need to fit the underlying model $N + 1$ times...
- ...and this needs to be done for each test sample!

Different extensions were proposed to overcome this problem:

- depending on the underlying model, in some special cases, **approximations** of full conformal prediction can be computed
- move away from transductive inference: **split conformal prediction** (a.k.a. inductive conformal prediction)

Split conformal prediction

- Split the training set in two to extract a **validation set** of size N_{val}, $(\boldsymbol{X}'_1, Y'_1), \ldots, (\boldsymbol{X}'_{N_{\text{val}}}, Y'_{N_{\text{val}}})$.
- Train the model on the new training set.
- Compute the threshold t on the validation set.

Split conformal predictors can be more easily expressed using solely nonconformity scores:

$$s : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$$

using a thresholding method, for a threshold t,

$$\Gamma(\boldsymbol{x}) = \{y \in \mathcal{Y} \mid s(\boldsymbol{x}, y) \leq t\}.$$

Typically, $s(\boldsymbol{x}, y) = 1 - \hat{p}(y \mid \boldsymbol{x})$.

Split conformal prediction

Under only an exchangeability assumption, we have a **guarantee on the coverage** by picking the threshold as an appropriate quantile of the nonconformity scores computed on the training data:

$$t = \text{Quantile}\left(\left(1 + \frac{1}{N_{\text{val}}}\right)\alpha; \left\{s(\boldsymbol{X}'_i, Y'_i)\right\}_{i=1}^{N_{\text{val}}}\right).$$

This guarantee holds **independently** of the model and of the nonconformity score.
However, by default, there is **no guarantee** on the informativeness of the predicted sets!

Summary

- A **lot of different formulations** of set-valued classification exist in the literature:
  - They have different properties and **are useful in different scenarios**.
  - However, it is not always clear which formulation to use: can be **confusing**.
- For both classification with reject option and set-valued classification, there is not a lot of literature on the connection with **uncertainty representation**.
- It could be interesting to use **both** a reject option with a set-valued predictor to treat differently aleatoric and epistemic uncertainty:
  - reject in presence of epistemic uncertainty
  - predict a set in presence of aleatoric uncertainty

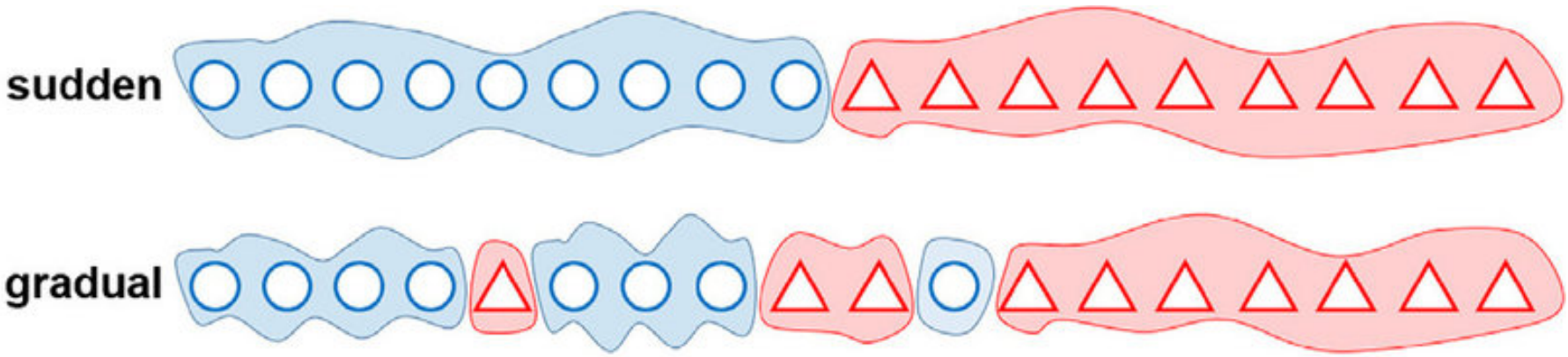
Overview of this talk

Overview of this talk

1. Introduction
2. Uncertainty in classification: representation and methods (40 minutes)
3. Coffee break
4. Classification with reject option (25 minutes)
5. Set-valued prediction methods (25 minutes)
6. Coffee break
7. Rejection and set-valued prediction with distribution shift (40 minutes)

What is distribution shift ?

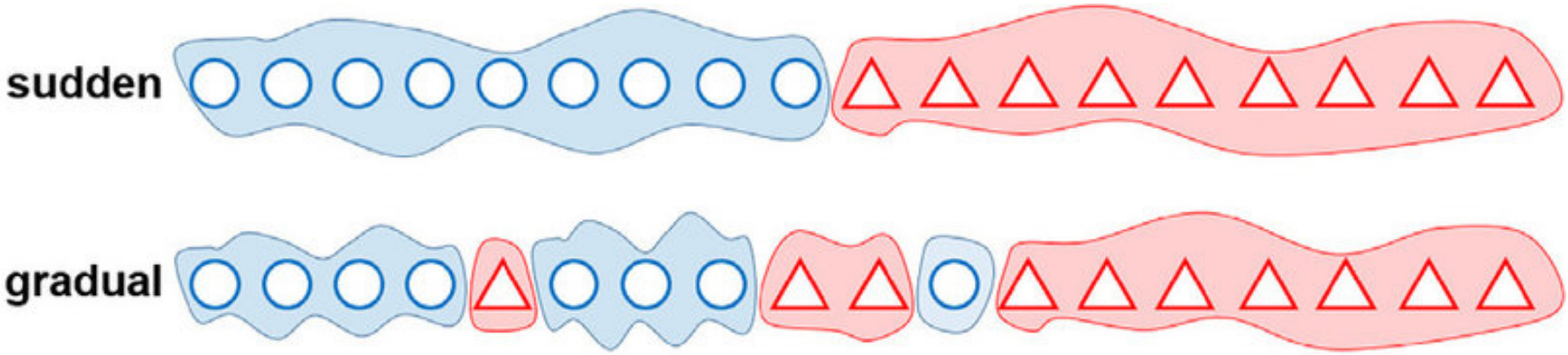
Dataset shift appears when **training and test joint distributions are different**, that is when $p_{\text{train}}(\boldsymbol{x}, y) \neq p_{\text{test}}(\boldsymbol{x}, y)$.



Abstention is needed in such cases, because a model cannot reliably classify instances that are very different from those seen during training.

What is distribution shift ?

Dataset shift appears when **training and test joint distributions are different**, that is when $p_{\text{train}}(\boldsymbol{x}, y) \neq p_{\text{test}}(\boldsymbol{x}, y)$.



Abstention is needed in such cases, because a model cannot reliably classify instances that are very different from those seen during training.

Main types of dataset shift:

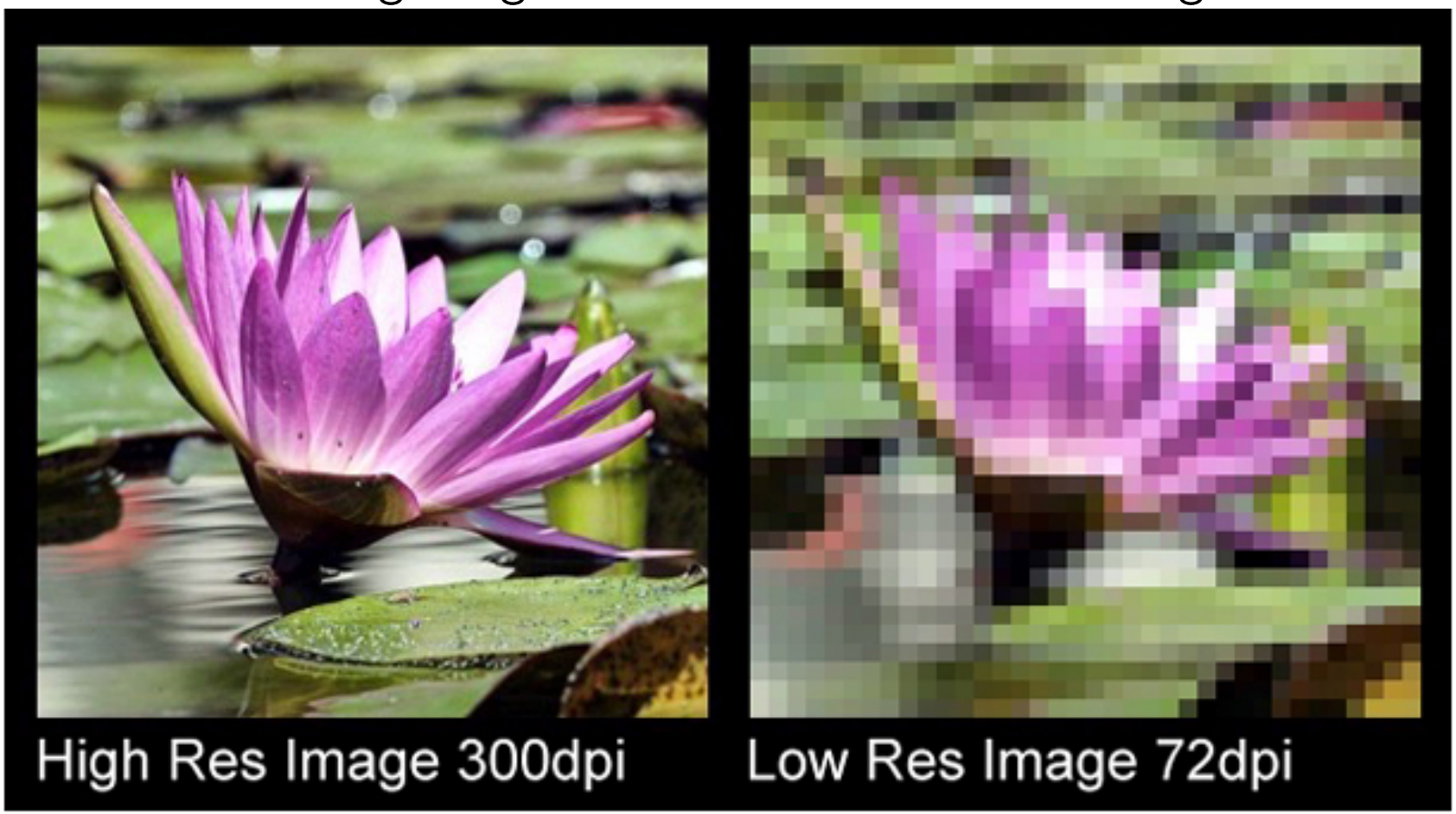
- Covariate shift
- Prior probability shift
- Concept shift/drift

Covariate shift

Happens when $p_{\text{train}}(y \mid \boldsymbol{x}) = p_{\text{test}}(y \mid \boldsymbol{x})$ and $p_{\text{train}}(\boldsymbol{x}) \neq p_{\text{test}}(\boldsymbol{x})$

Training image

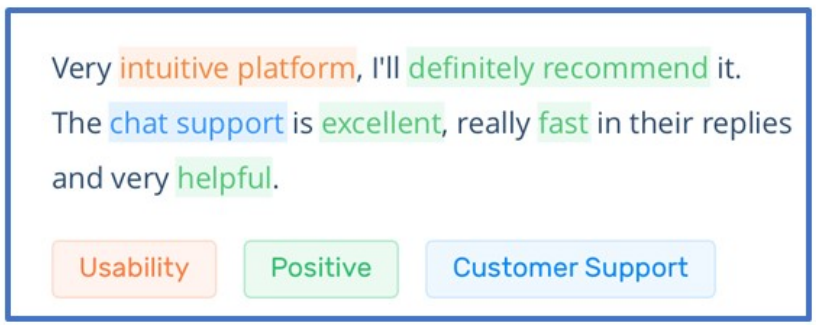
Test image



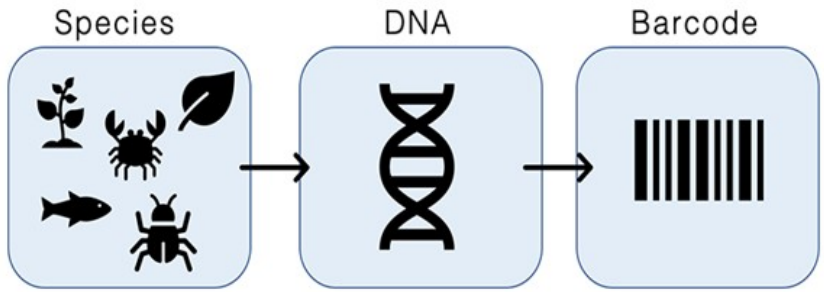
Prior probability shift

Happens when $p_{\text{train}}(\boldsymbol{x} \mid y) = p_{\text{test}}(\boldsymbol{x} \mid y)$ and $p_{\text{train}}(y) \neq p_{\text{test}}(y)$

Computer vision



NLP tasks



Bioinformatics

Picture taken from: (1) Sharon Yixuan Li (2) MonkeyLearn

(3) https://bitesizebio.com/44108/whats-that-organism-using-dna-barcoding-for-species-identification/

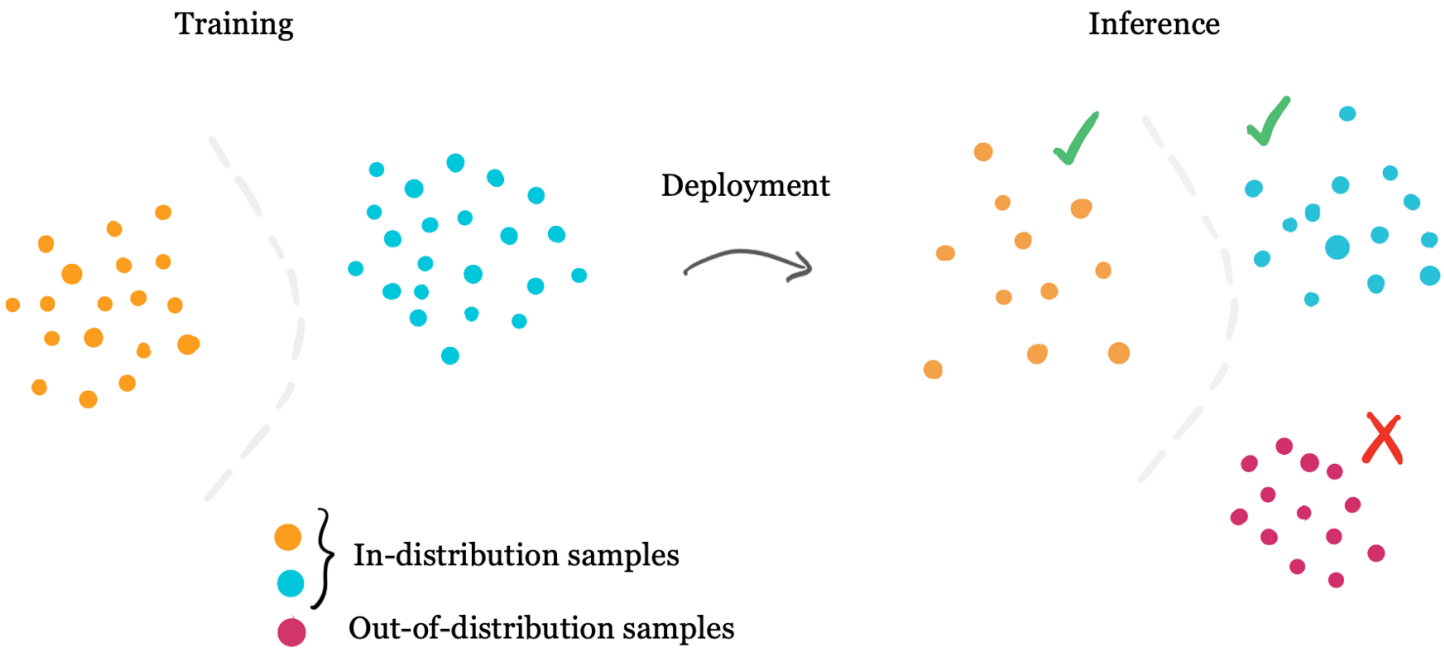
Concept shift

Happens when $p_{\text{train}}(y \mid \boldsymbol{x}) \neq p_{\text{test}}(y \mid \boldsymbol{x})$ and $p_{\text{train}}(\boldsymbol{x}) = p_{\text{test}}(\boldsymbol{x})$

Source: Beymour Consulting

Out-of-distribution (OOD) detection

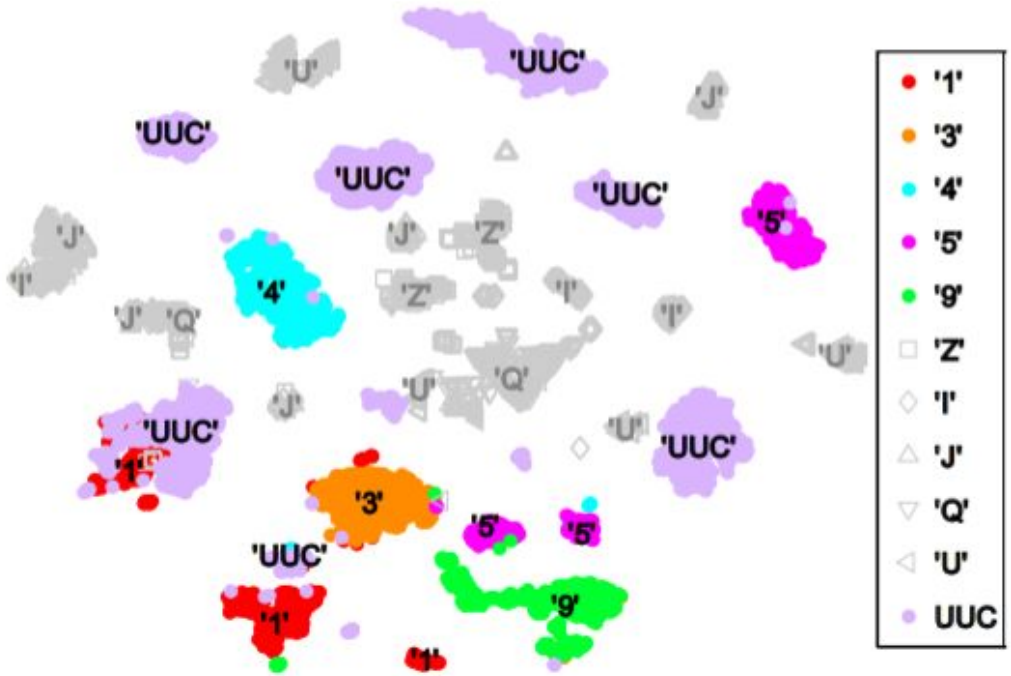
- Detection of a test sample $\boldsymbol{x}$ that does not belong to the training distribution $p_{\text{train}}(\boldsymbol{x}, y)$
- Papers that present OOD methods rarely specify what type of dataset shift they consider
- In experiments: often prior probability shift, sometimes also covariate shift, almost never concept shift



Picture taken from: https://medium.com/geekculture/out-of-distribution-detection-in-medical-ai-b638b385c2a3

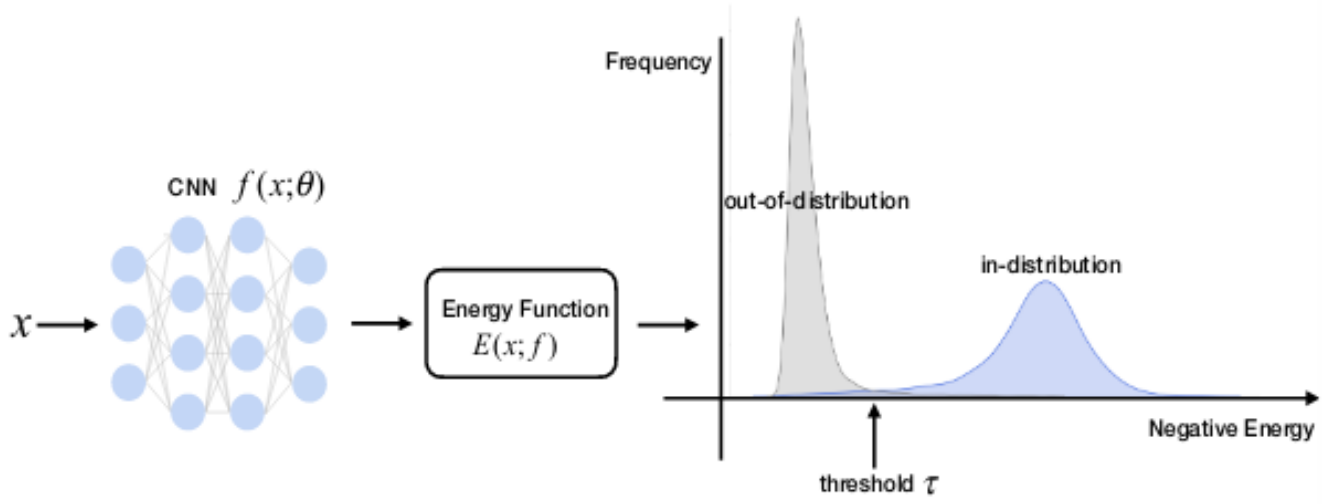
Open set and open world recognition

- Detection of test samples corresponding to classes with no training samples
- Only detection of such samples when no side information is available (in the other case, classification via zero-shot learning)
- Often the prior probability shift setting is considered, but might be needed as well in the i.i.d. setting



Picture taken from: https://arxiv.org/pdf/1811.08581.pdf

Methodology of a typical OOD detection paper

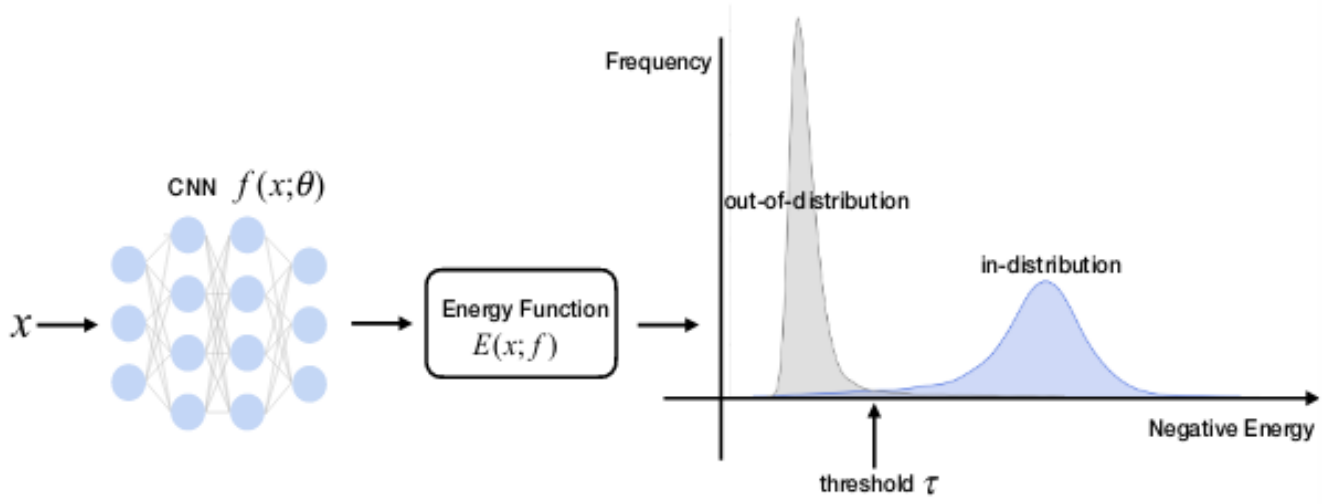


Example taken from Liu et al. Energy-based Out-of-distribution Detection, Neurips 2020.

Use training data to learn

1. a latent representation $f_\theta(\boldsymbol{x})$

Methodology of a typical OOD detection paper

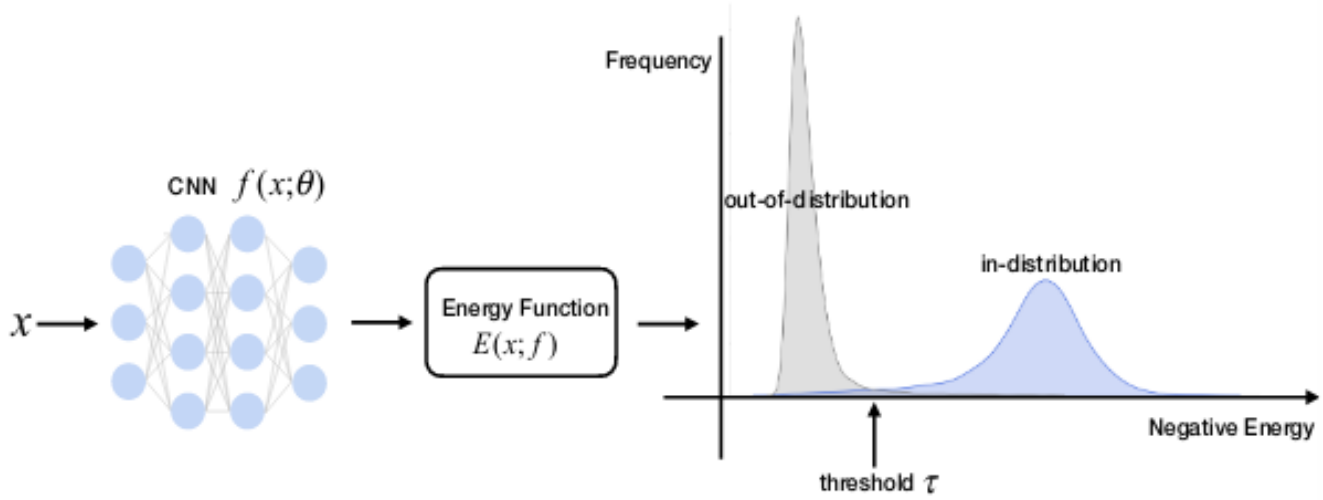


Example taken from Liu et al. Energy-based Out-of-distribution Detection, Neurips 2020.

Use training data to learn

1. a latent representation $\mathrm{f}_\theta(\boldsymbol{x})$
2. a scoring function $s(\mathrm{f}_\theta(\boldsymbol{x}))$ that quantifies how confident we are whether a test point $\boldsymbol{x}$ is out-of-distribution

Methodology of a typical OOD detection paper

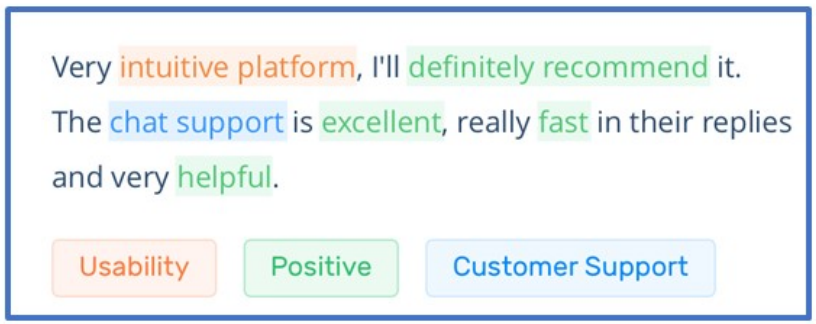


Example taken from Liu et al. Energy-based Out-of-distribution Detection, Neurips 2020.

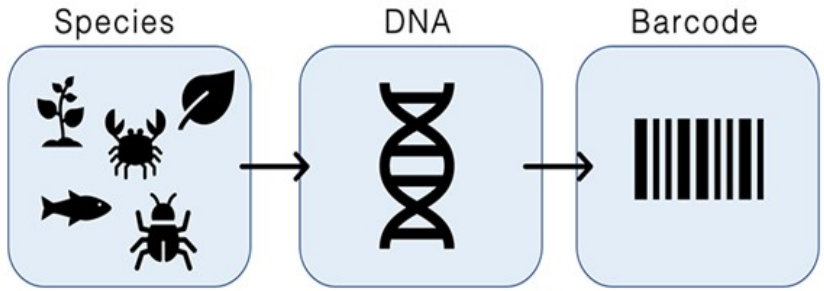
Use training data to learn

1. a latent representation $\mathrm{f}_\theta(\boldsymbol{x})$
2. a scoring function $s(\mathrm{f}_\theta(\boldsymbol{x}))$ that quantifies how confident we are whether a test point $\boldsymbol{x}$ is out-of-distribution
   - Explicit usage of an OOD class
   - OOD scores derived from $p(y \mid \boldsymbol{x}, \theta)$
   - OOD scores derived from generative models $p(\boldsymbol{x} \mid \theta)$, ignoring labels
   - OOD scores derived from generating models $p(\boldsymbol{x} \mid y, \theta)$, using labels

Typical application domains



NLP tasks



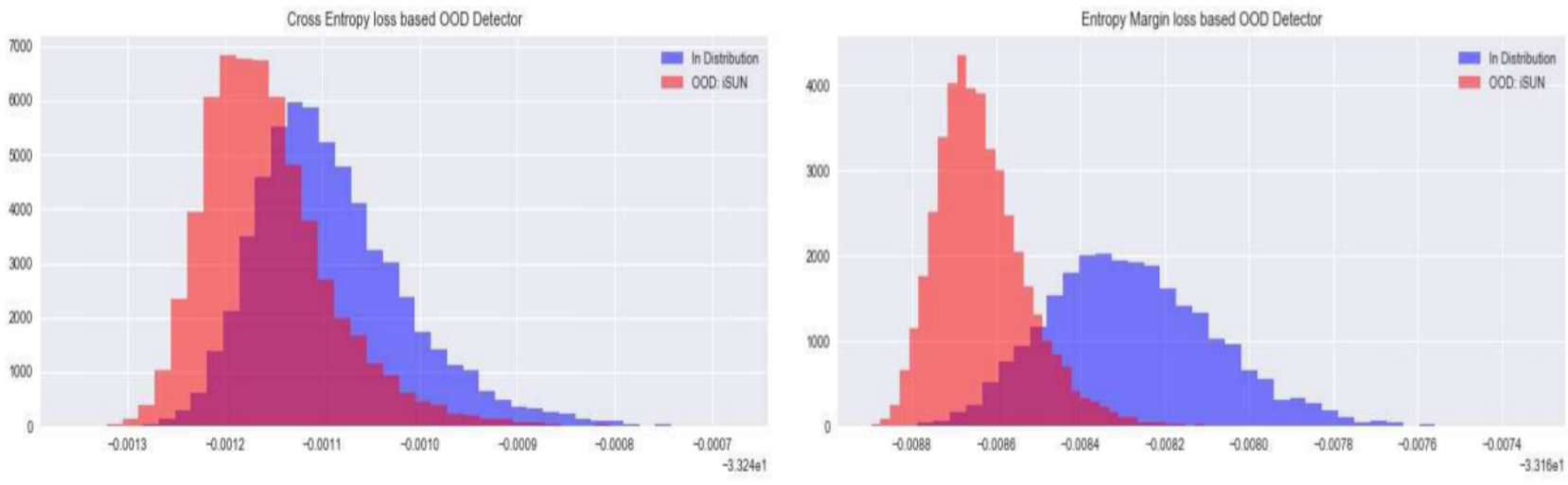
Bioinformatics

Picture taken from: (1) Sharon Yixuan Li (2) MonkeyLearn

(3) https://bitesizebio.com/44108/whats-that-organism-using-dna-barcoding-for-species-identification/

Experimental protocol of a typical OOD detection paper

1. Often the prior probability shift setting
2. Often OOD exposure during training, hyperparameter tuning or threshold selection
3. Define an evaluation metric:
   - Without threshold selection, e.g. ROC AUC, PR AUC, etc.
   - With threshold selection, e.g. Accurarcy, FPR, etc.



(a) Cross entropy loss. (b) Margin entropy loss.

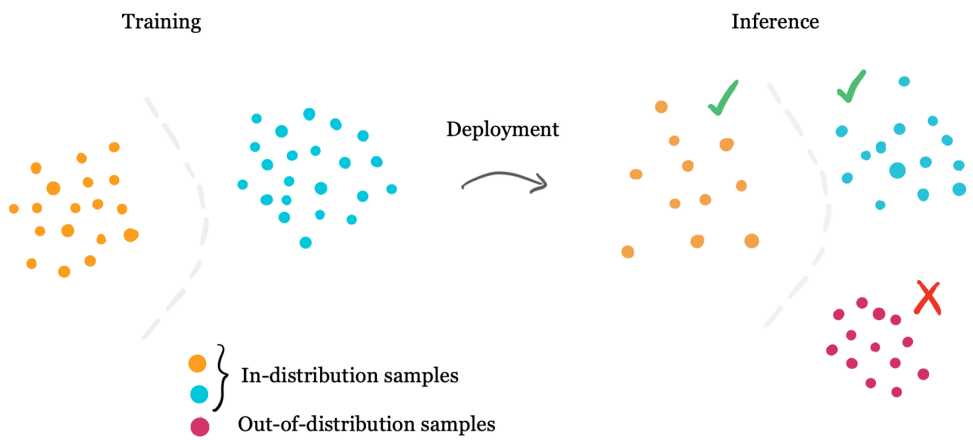
Example taken from Vyas et al., Out-of-Distribution Detection Using an Ensemble of Self Supervised Leave-out Classifiers, ECCV 2018.

Explicit modelling of an OOD class[26],[27],[28]

- In C-class classification, define a class $\bar{y}_{C+1}$ that collects all possible OOD data and use a standard softmax neural network:

$$\sum_{j=1}^{C+1} p(y = \bar{y}_j \mid \boldsymbol{x}, \theta) = 1$$

- OOD data must be collected for training
- Difficult to anticipate on future OOD samples that deviate from "training" OOD set



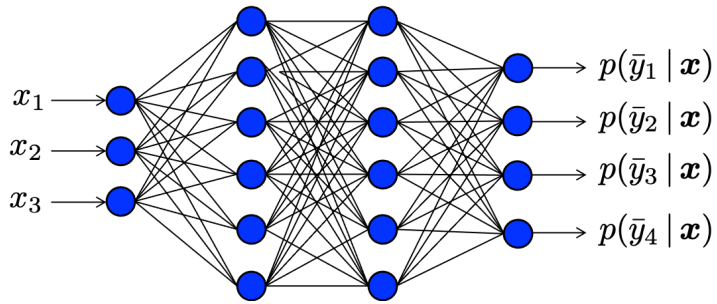
[26] Neal et al., Open set learning with counterfactual images, ECCV 2018

[27] Mohseni et al., Self-Supervised Learning for Generalizable Out-of-Distribution Detection, AAAI 2020

[28] Thulasidasan et al., A Simple and Effective Baseline for Out-of-Distribution Detection using Abstention, OpenReview 2021

OOD scores derived from $p(y \mid \boldsymbol{x}, \theta)$

- Fit a traditional softmax-style discriminative neural network:



- Mode[29]:

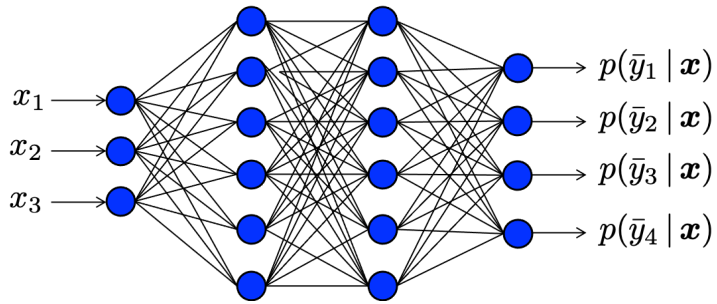
$$s(\boldsymbol{x}) = \max_{j \in \{1,\ldots,C\}} p(y = \bar{y}_j \mid \boldsymbol{x}, \theta)$$

[29] Hendrycks and Gimpel, A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks, ICLR 2017

[30] Lakshminarayanan et al., Simple and scalable predictive uncertainty estimation using deep ensembles, Neurips 2017

OOD scores derived from $p(y \mid \boldsymbol{x}, \theta)$

- Fit a traditional softmax-style discriminative neural network:



- Mode[29]:

$$s(\boldsymbol{x}) = \max_{j \in \{1,\ldots,C\}} p(y = \bar{y}_j \mid \boldsymbol{x}, \theta)$$

- Cross-entropy[30]:

$$s(\boldsymbol{x}) = \sum_{j=1}^{C} p(y = \bar{y}_j \mid \boldsymbol{x}, \theta) \log p(y = \bar{y}_j \mid \boldsymbol{x}, \theta)$$

[29] Hendrycks and Gimpel, A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks, ICLR 2017

[30] Lakshminarayanan et al., Simple and scalable predictive uncertainty estimation using deep ensembles, Neurips 2017

Outlier exposure in discriminative models

- Improvements reported when **OOD data is used during training** and/or hyperparameter tuning
- Incorporating outlier exposure in the loss function[31]:

$$\mathbb{E}_{(\boldsymbol{x},y)\sim p_{\text{train}}}\left[\ell(\mathsf{f}_\theta(\boldsymbol{x}),y)+\lambda\mathbb{E}_{(\boldsymbol{x}')\sim p_{\text{train}}}\ell_{OE}(\mathsf{f}_\theta(\boldsymbol{x}),\mathsf{f}_\theta(\boldsymbol{x}'))\right]$$

[31] Hendryckx et al., Deep anomaly detection with outlier exposure, ICLR 2019

[32] Liang et al. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks, ICLR 2018

Outlier exposure in discriminative models

- Improvements reported when **OOD data is used during training** and/or hyperparameter tuning
- Incorporating outlier exposure in the loss function[31]:

$$\mathbb{E}_{(\boldsymbol{x},y)\sim p_{\text{train}}}\left[\ell(\mathsf{f}_\theta(\boldsymbol{x}),y)+\lambda\mathbb{E}_{(\boldsymbol{x}')\sim p_{\text{train}}}\ell_{OE}(\mathsf{f}_\theta(\boldsymbol{x}),\mathsf{f}_\theta(\boldsymbol{x}'))\right]$$

- Tuning temperature using OOD validation data[32]:

$$p(y=\bar{y}_j\,|\,\boldsymbol{x},\theta,T)=\frac{\exp(f_j(\boldsymbol{x},\theta)/T)}{\sum_{s=1}^{C}\exp(f_s(\boldsymbol{x},\theta)/T)}$$

[31] Hendryckx et al., Deep anomaly detection with outlier exposure, ICLR 2019

[32] Liang et al. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks, ICLR 2018

Outlier exposure in discriminative models

- Improvements reported when **OOD data is used during training** and/or hyperparameter tuning
- Incorporating outlier exposure in the loss function[31]:

$$\mathbb{E}_{(\boldsymbol{x},y)\sim p_{\text{train}}}\left[\ell(\mathsf{f}_\theta(\boldsymbol{x}),y)+\lambda\mathbb{E}_{(\boldsymbol{x}')\sim p_{\text{train}}}\ell_{OE}(\mathsf{f}_\theta(\boldsymbol{x}),\mathsf{f}_\theta(\boldsymbol{x}'))\right]$$

- Tuning temperature using OOD validation data[32]:

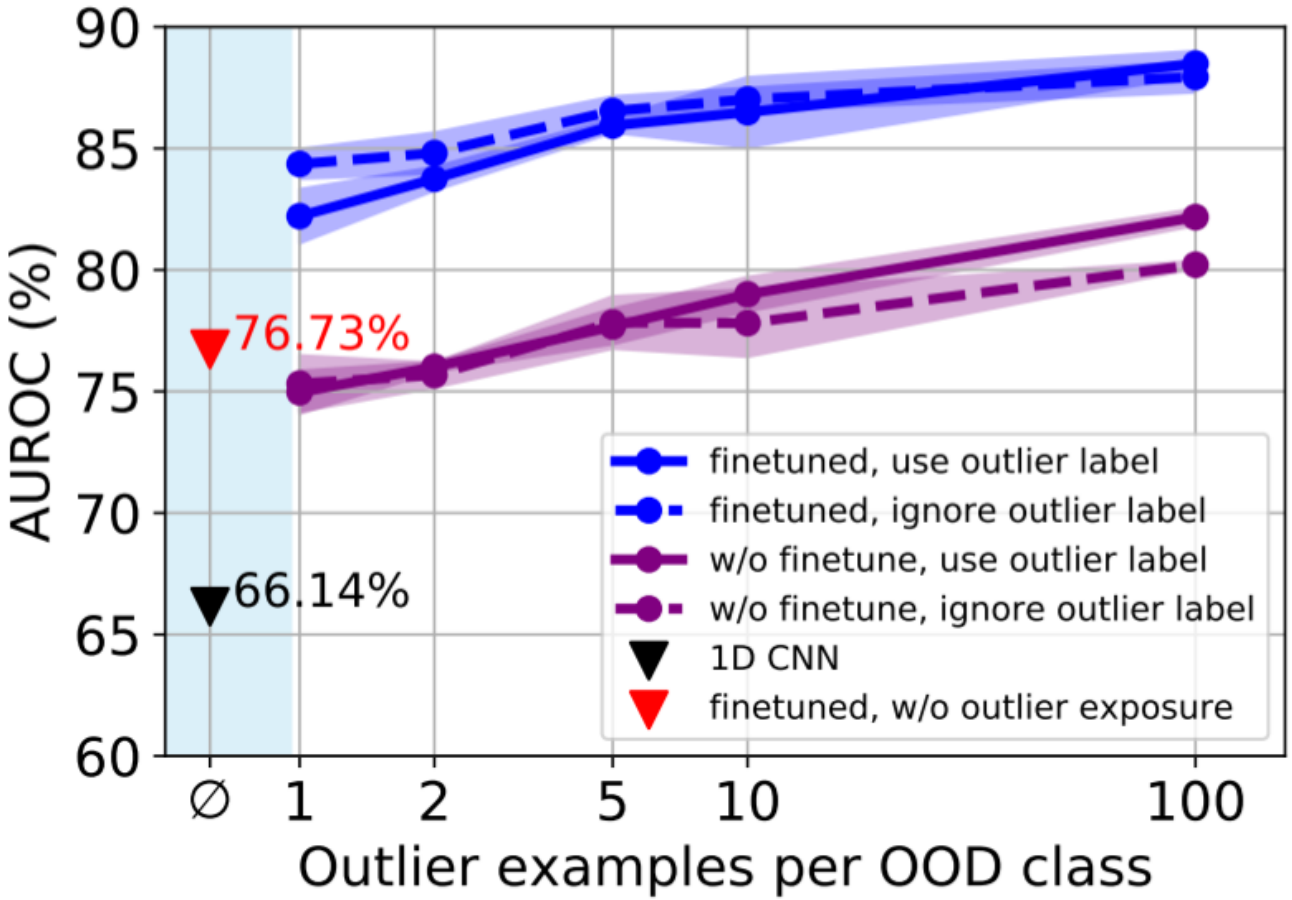
$$p(y=\bar{y}_j\,|\,\boldsymbol{x},\theta,T)=\frac{\exp(f_j(\boldsymbol{x},\theta)/T)}{\sum_{s=1}^{C}\exp(f_s(\boldsymbol{x},\theta)/T)}$$

- Typical **but questionable** assumption: uniform distribution $p(y\,|\,\boldsymbol{x},\theta)$ when $\boldsymbol{x}$ is OOD

[31] Hendryckx et al., Deep anomaly detection with outlier exposure, ICLR 2019

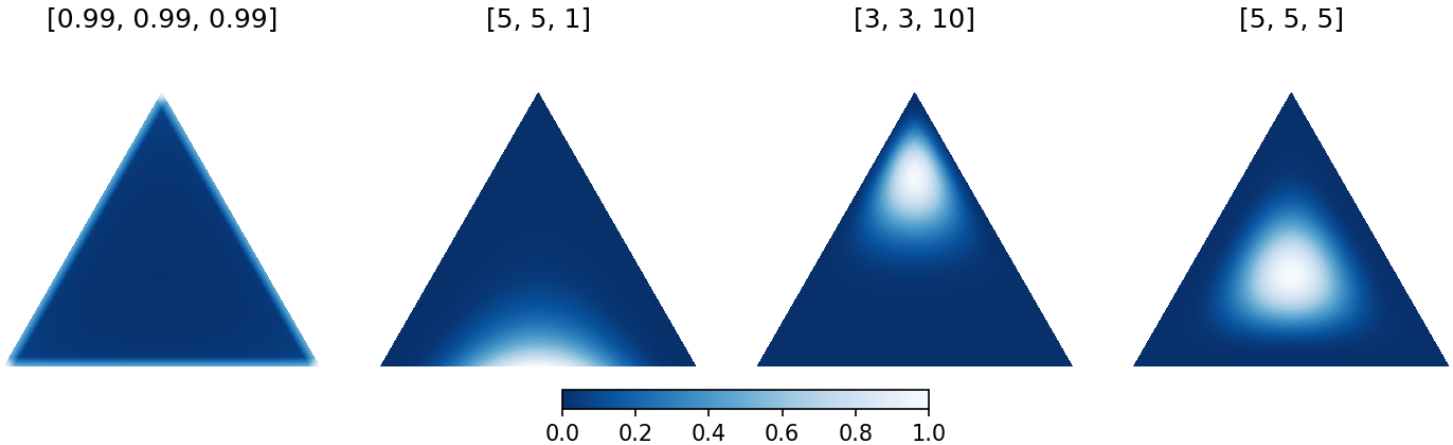
[32] Liang et al. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks, ICLR 2018

Outlier exposure usually leads to performance gains[33]



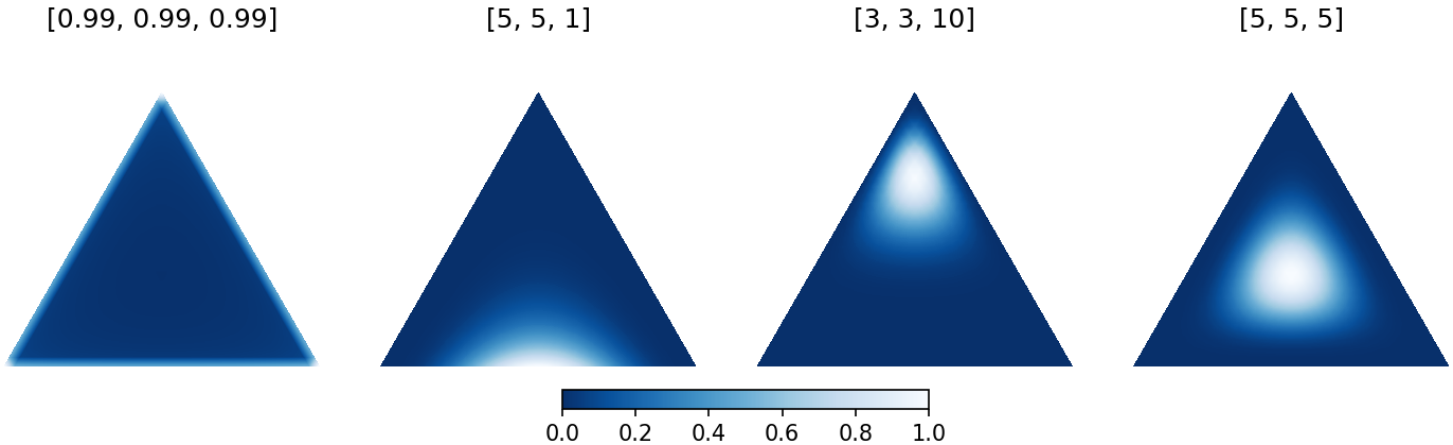
[33] Fort et al., Exploring the Limits of Out-of-Distribution Detection, Arxiv 2021.

OOD scores derived from Dirichlet distributions



- For every $\boldsymbol{x}_q$ assume a Dirichlet distribution $\mathrm{Dir}(\boldsymbol{\alpha}^{(q)})$

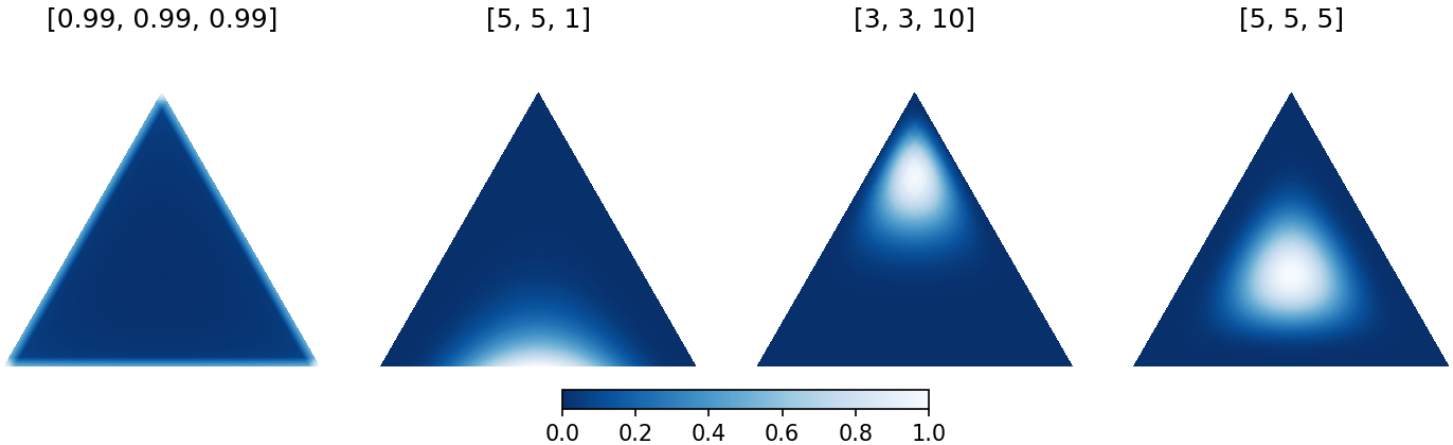
OOD scores derived from Dirichlet distributions



- For every $\boldsymbol{x}_q$ assume a Dirichlet distribution $\mathrm{Dir}(\boldsymbol{\alpha}^{(q)})$
- Model $\boldsymbol{\alpha}^{(q)}$ as the output layer of a neural network

$$p(y = \bar{y}_j \mid \boldsymbol{x}_q) = \frac{\alpha_j^{(q)}}{\sum_{s=1}^{C} \alpha_s^{(q)}} \qquad \hat{y}_q = \underset{s=1,\ldots,C}{\arg\max}\, p(y = \bar{y}_s \mid \boldsymbol{x}_q)$$

OOD scores derived from Dirichlet distributions



- For every $\boldsymbol{x}_q$ assume a Dirichlet distribution $\mathrm{Dir}(\boldsymbol{\alpha}^{(q)})$
- Model $\boldsymbol{\alpha}^{(q)}$ as the output layer of a neural network

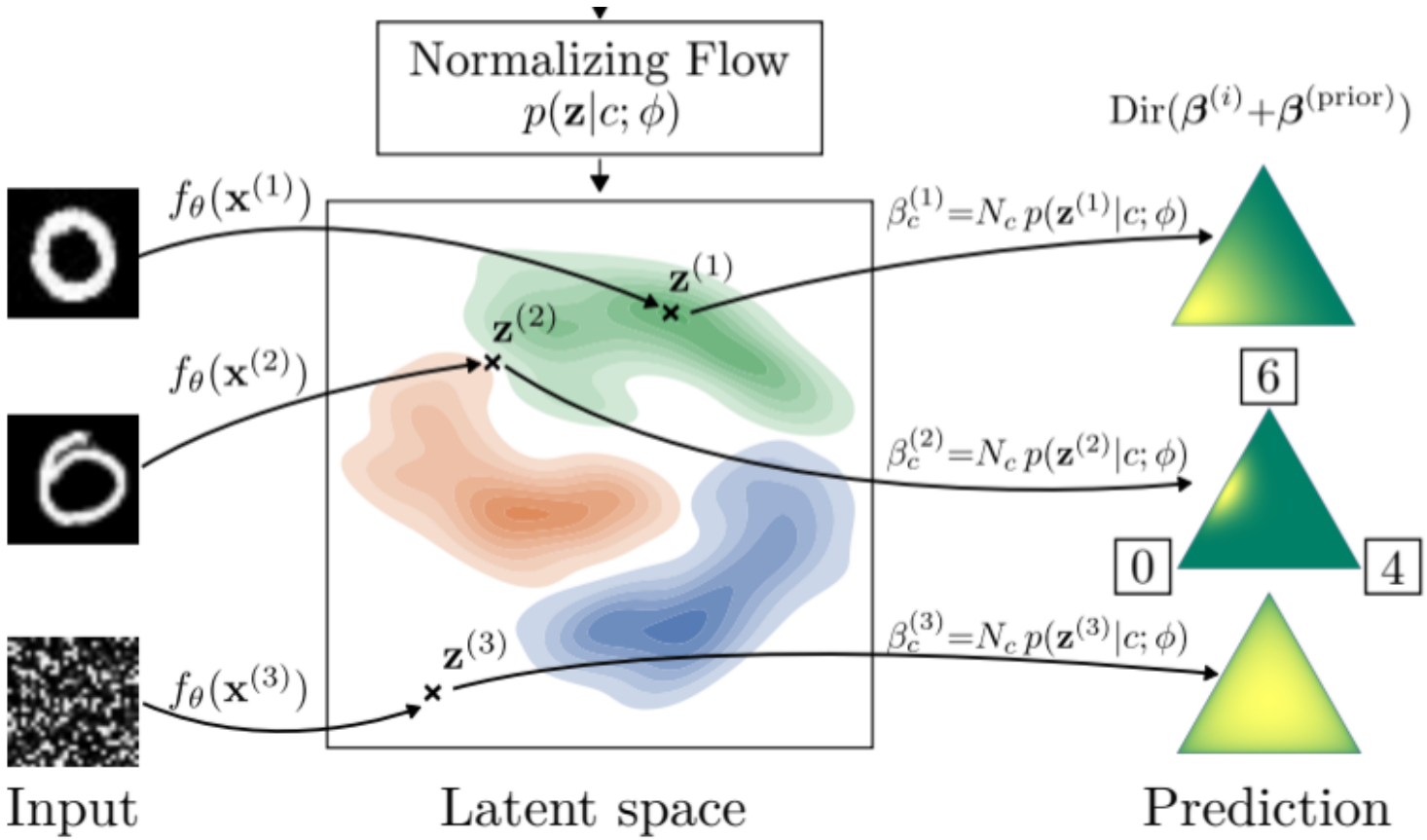
$$p(y = \bar{y}_j \mid \boldsymbol{x}_q) = \frac{\alpha_j^{(q)}}{\sum_{s=1}^{C} \alpha_s^{(q)}} \qquad \hat{y}_q = \underset{s=1,\ldots,C}{\arg\max}\, p(y = \bar{y}_s \mid \boldsymbol{x}_q)$$

- OOD detection via pseudo counts: $s(\boldsymbol{x}_q) = \sum_{s=1}^{C} \alpha_s^{(q)}$
- Alternative scoring functions: differential entropy, mutual information

Comparing various Dirichlet-based methods[34]

Table 1. Summary of DBU models. Further details on the loss functions are provided in the appendix.

	$\alpha^{(i)}$-parametrization	Loss	OOD training data	Ensemble training	Density estimation
PostNet	$f_\theta(\boldsymbol{x}^{(i)}) = \mathbf{1} + \boldsymbol{\alpha}^{(i)}$	Bayesian loss	No	No	Yes
PriorNet	$f_\theta(\boldsymbol{x}^{(i)}) = \boldsymbol{\alpha}^{(i)}$	Reverse KL	Yes	No	No
DDNet	$f_\theta(\boldsymbol{x}^{(i)}) = \boldsymbol{\alpha}^{(i)}$	Dir. Likelihood	No	Yes	No
EvNet	$f_\theta(\boldsymbol{x}^{(i)}) = \mathbf{1} + \boldsymbol{\alpha}^{(i)}$	Expected MSE	No	No	No



[34] Kopetzki et al. Evaluating Robustness of Predictive Uncertainty Estimation: Are Dirichlet-based Models Reliable? ICML 2021

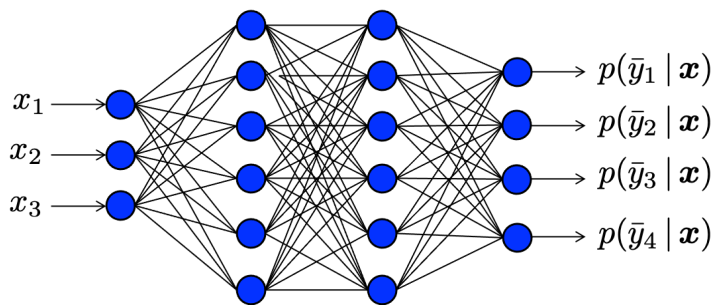
Modelling $p(\boldsymbol{x})$ using $p(y \mid \boldsymbol{x})$[35]

- Scoring function is energy of softmax network:

$$s(\boldsymbol{x}) = E(\boldsymbol{x}, \mathrm{f}_\theta) = T \cdot \log \sum_{j=1}^{C} e^{f_j(\boldsymbol{x})/T}$$

- Thus, **no normalization step** as softmax output
- Training objective: high energy for IND data and low energy for OOD data
- Connection with generative models:

$$p(\boldsymbol{x}) = \frac{E(\boldsymbol{x}, \mathrm{f}_\theta)/T}{\int_{\mathcal{X}} E(\boldsymbol{x}, \mathrm{f}_\theta)/T}$$

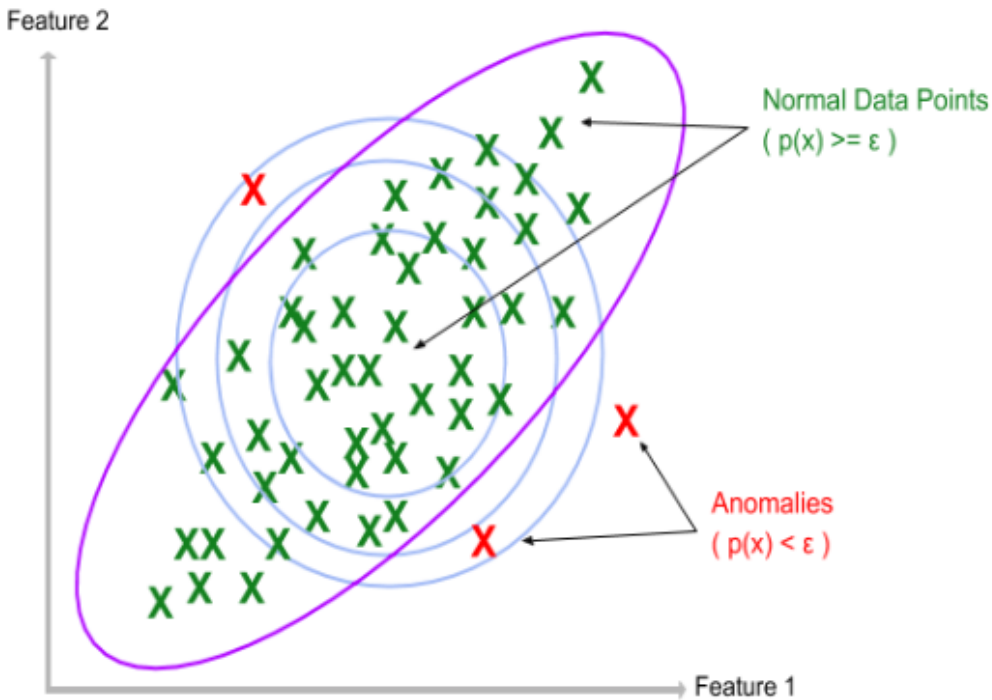


[35] Liu et al., Energy-based out-of-distribution detection, Neurips 2020

OOD scores derived from $p(\boldsymbol{x} \mid \theta)$

Outlier and anomaly detection

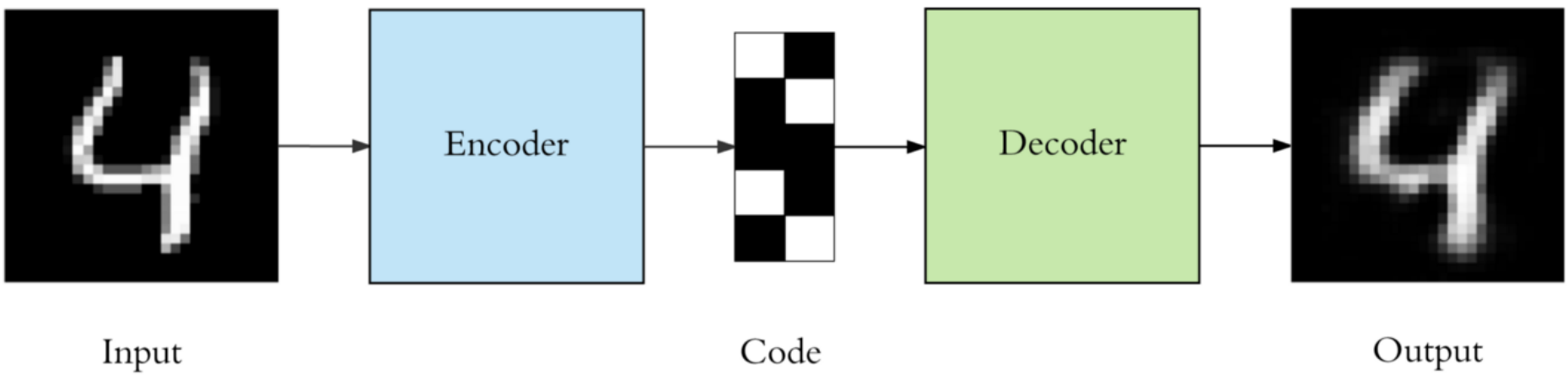
- Detection of a test sample $\boldsymbol{x}$ that does not belong to the training distribution $p_{\text{train}}(\boldsymbol{x})$
- Often training data is unlabeled, goal is to separate "normal" from "anomalous" data
- Very general terms, not always dataset shift



Picture taken from: https://towardsdatascience.com/introduction-to-anomaly-detection-c651f38ccc32

OOD scores derived from $p(\boldsymbol{x} \mid \theta)$: Overview of methods

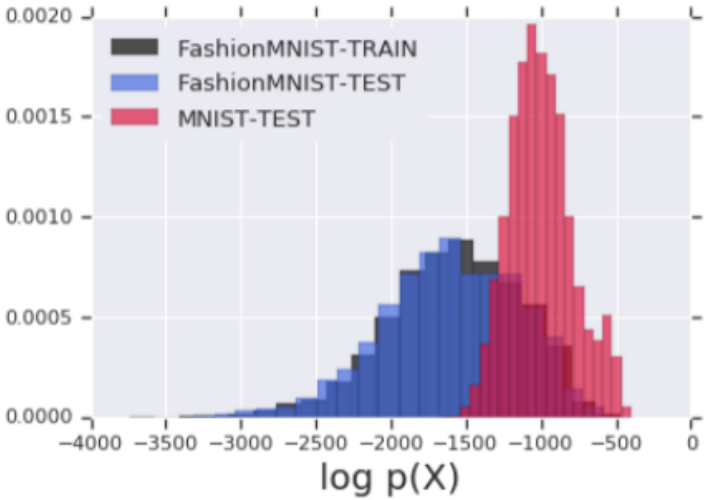
- Traditional density-based methods, e.g. Gaussian mixtures, Kernel density estimation, etc.
- Traditional distance-based methods, e.g. KNN, K-means, one-class SVMs, etc.
- Deep generative models: variational auto-encoders, normalizing flows, autoregressive models, etc.



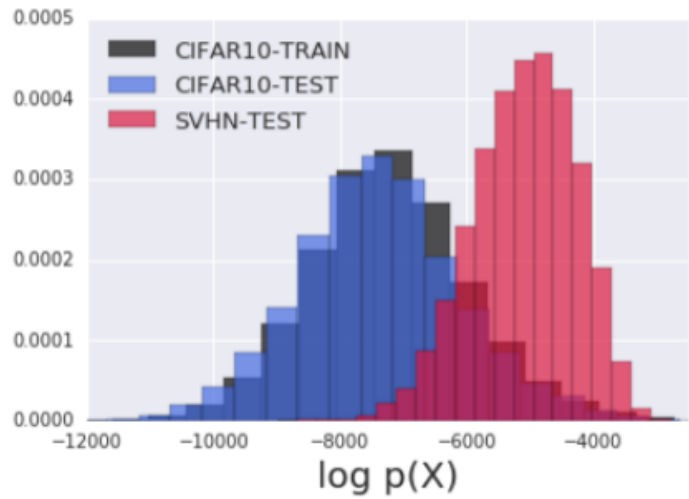
Picture taken from: https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

Some negative results for deep generative models

- Often higher likelihoods for OOD data than IND data[36],[37],[38]
- Observed for variational auto-encoders, normalizing flows and autoregressive models



(a) Train on FashionMNIST, Test on MNIST



(b) Train on CIFAR-10, Test on SVHN

[36] Nalisnick et al. Do deep generative models know what they don't know?, ICLR 2019

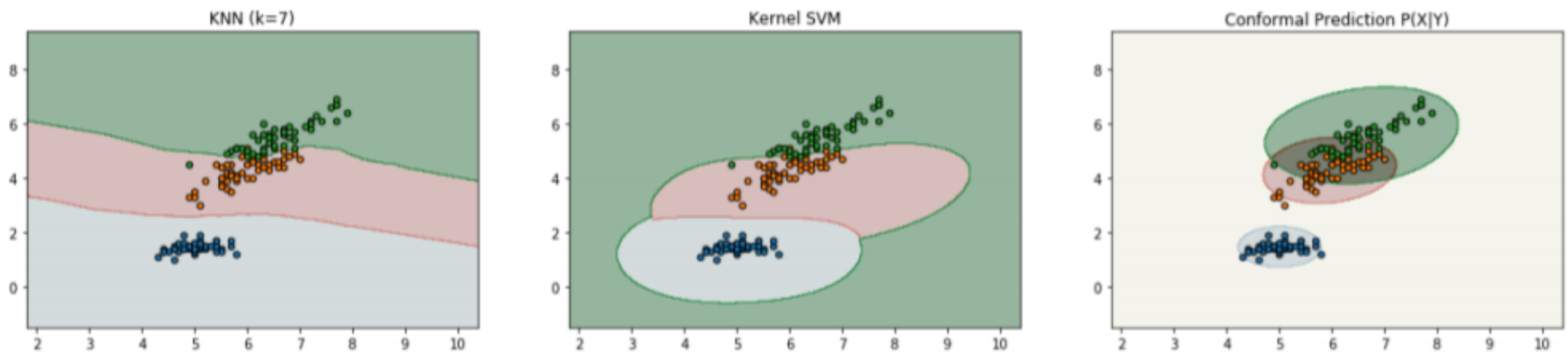
[37] Kirichenko et al., Why Normalizing Flows Fail to Detect Out-of-Distribution Data?, Neurips 2020

[38] Understanding failures in OOD detection with deep generative models, ICML 2021

OOD scores derived from $p(\boldsymbol{x} \mid y, \theta)$: KDE[39]

1. Fit a discrinative softmax-style deep neural network to obtain $\mathrm{f}_\theta(\boldsymbol{x})$
2. Fit $p(\mathrm{f}_\theta(\boldsymbol{x})|y = \bar{y}_j)$ via kernel density estimation
3. Select a threshold per class without OOD exposure using the $1 - \alpha$ quantile:

$$\hat{t}(j) = \sup \left\{ t : \sum_{i=1}^{N_j} I(p(\mathrm{f}_\theta(\boldsymbol{x}_i)|y = \bar{y}_j) \geq t) \geq 1 - \alpha \right\}$$



[39] Hechtlinger et al., Cautious Deep Learning, Arxiv 2019.

OOD scores derived from $p(\boldsymbol{x} \mid y, \theta)$: LDA [40]

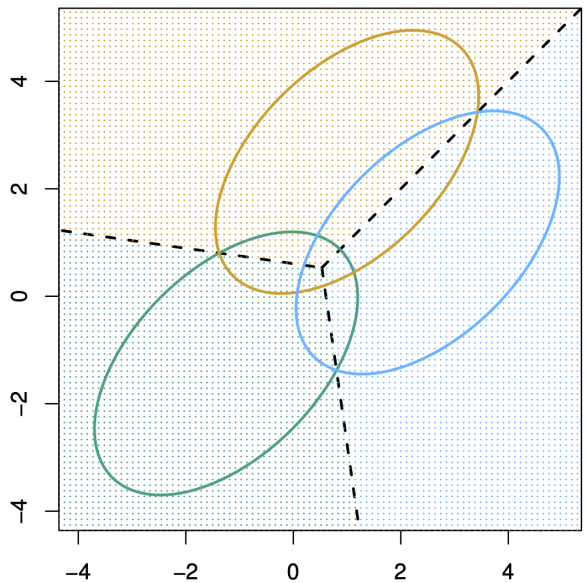
The standard recipe:

1. Fit a discrinative softmax-style deep neural network
2. Apply linear discriminant analysis in the learned latent space

$$p(\mathrm{f}_\theta(\boldsymbol{x}) \mid y = \bar{y}_j, \theta) = \mathcal{N}(\mu_j, \Sigma)$$

3. Scoring function obtained via Mahalanobis distance

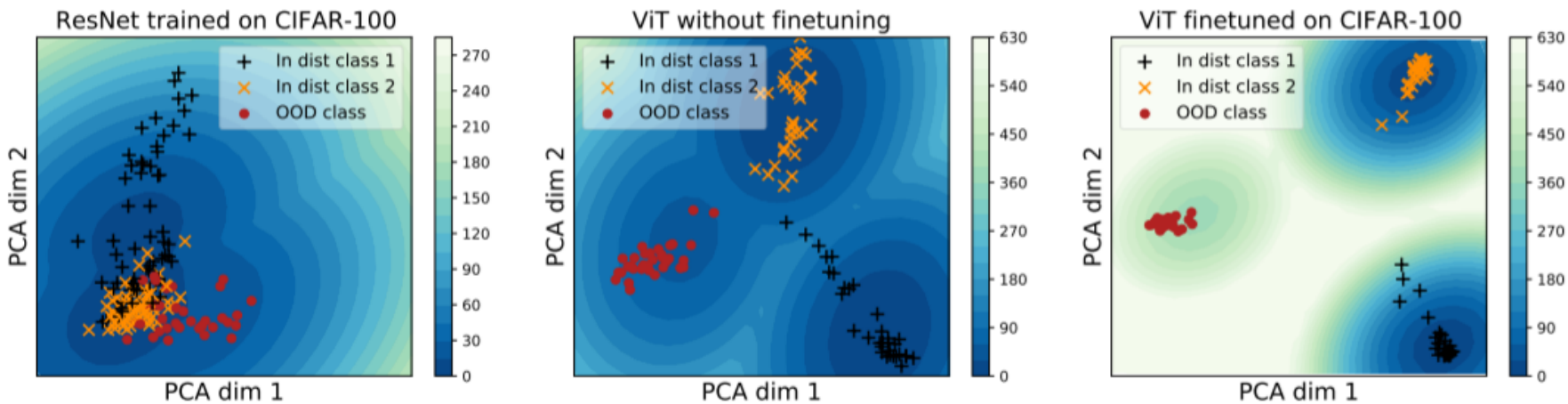
$$s(\boldsymbol{x}) = \min_{j=1,\ldots,C} (\mathrm{f}_\theta(\boldsymbol{x}) - \mu_j)^T \hat{\Sigma}^{-1} (\mathrm{f}_\theta(\boldsymbol{x}) - \mu_j)$$



[40] Lee et al. A simple unified framework for detecting out-of-distribution samples and adversarial attacks, Neurips 2018.

Feature learning for LDA[41]

- The quality of the learning embedding $f_\theta(\boldsymbol{x})$ is very important
- SOTA results for **near-OOD tasks** using vision transformers and finetuning, e.g. CIFAR-100 in-distribution and CIFAR-10 out-of-distribution



[41] Fort et al., Exploring the Limits of Out-of-Distribution Detection, Arxiv 2021.

OOD scores derived from $p(\boldsymbol{x} \mid y, \theta)$: RBF output layer[42]

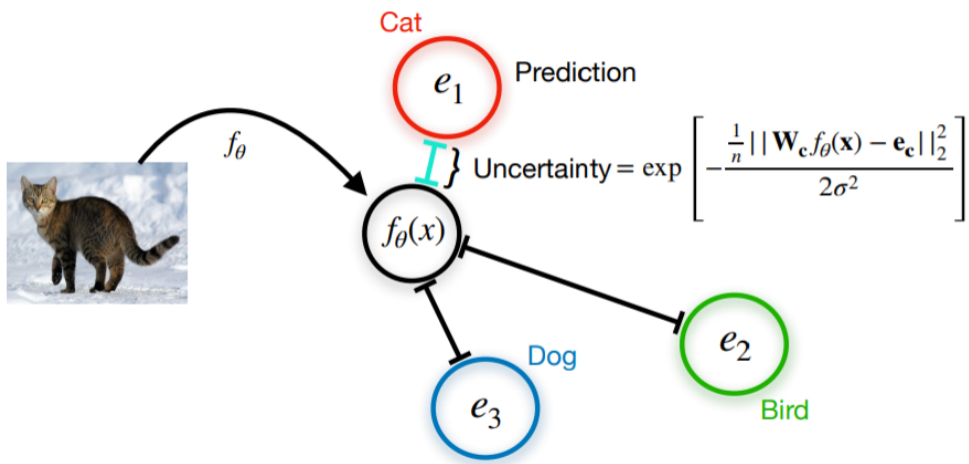
1. Consider a centroid $\boldsymbol{e}_j$ for class $\bar{y}_j$ in a latent space $\mathrm{f}_\theta(\boldsymbol{x})$
2. Similarity between a new data point $\boldsymbol{x}$ and the centroids as follows:

$$K(f_\theta(\boldsymbol{x}), \boldsymbol{e}_j) = \exp\left[\frac{\frac{1}{N}||W_j \mathrm{f}_\theta(\boldsymbol{x}) - \boldsymbol{e}_j||_2^2}{2\sigma^2}\right]$$

3. Scoring function:

$$s(\boldsymbol{x}) = \max_{j=1,\ldots,C} K(f_\theta(\boldsymbol{x}), \boldsymbol{e}_j)$$

4. Optimize θ, $W_1, \ldots, W_C$ and $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_C$ using binary cross-entropy loss



[42] Van Amersfoort et al., Uncertainty estimation using a single deep deterministic neural network, ICML 2020.

Reject option in case of covariate shift

- Much less often studied in the literature[43],[44]
- Typically **a closed world** assumption
- Hence, more focus on calibration, accuracy and discriminative models
- For example, Brier score using one-hot-encoded label vector $\boldsymbol{y}^*$:

$$BS(\boldsymbol{x}, y) = \sum_{j=1}^{C} (y_j^* - p(y = \bar{y}_j \mid \boldsymbol{x}, \theta))^2$$

(a) Rotated MNIST (b) Translated MNIST (c) Confidence vs Acc Rotated 60°

[43] Ovadia et al. Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift, Neurips 2019

[44] Krueger et al., OOD Generalization via Risk Extrapolation, ICML 2021

Reject option in case of concept drift

- This setting is much less studied in the literature

Reject option in case of concept drift

- This setting is much less studied in the literature
- Some **transfer learning** and **domain adaptation** methods can be applied

$$\begin{aligned} \text{Source domain:} \quad & (\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_N, y_{N_{tr}}) \sim p_{\text{train}}(\boldsymbol{x}, y) \\ \text{Target domain:} \quad & (\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_N, y_{N_{te}}) \sim p_{\text{test}}(\boldsymbol{x}, y) \\ \text{Assumption:} \quad & N_{tr} >> N_{te} \end{aligned}$$

- However, no focus on reject options in such papers

Reject option in case of concept drift

- This setting is much less studied in the literature
- Some **transfer learning** and **domain adaptation** methods can be applied

$$\begin{aligned} \text{Source domain:} \quad & (\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_N, y_{N_{tr}}) \sim p_{\text{train}}(\boldsymbol{x}, y) \\ \text{Target domain:} \quad & (\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_N, y_{N_{te}}) \sim p_{\text{test}}(\boldsymbol{x}, y) \\ \text{Assumption:} \quad & N_{tr} >> N_{te} \end{aligned}$$

- However, no focus on reject options in such papers
- More focus in some data stream classification papers

Reject option in data stream classification[45]

1. Estimate $p(y_t \mid \boldsymbol{x}_t, y_{t-1}, ..., t_{t-T})$ with an online learning method

[45] Goepfert et al., Mitigating the Adverse Effects of Concept Drift via the Application of Reject Option, ICANN 2018

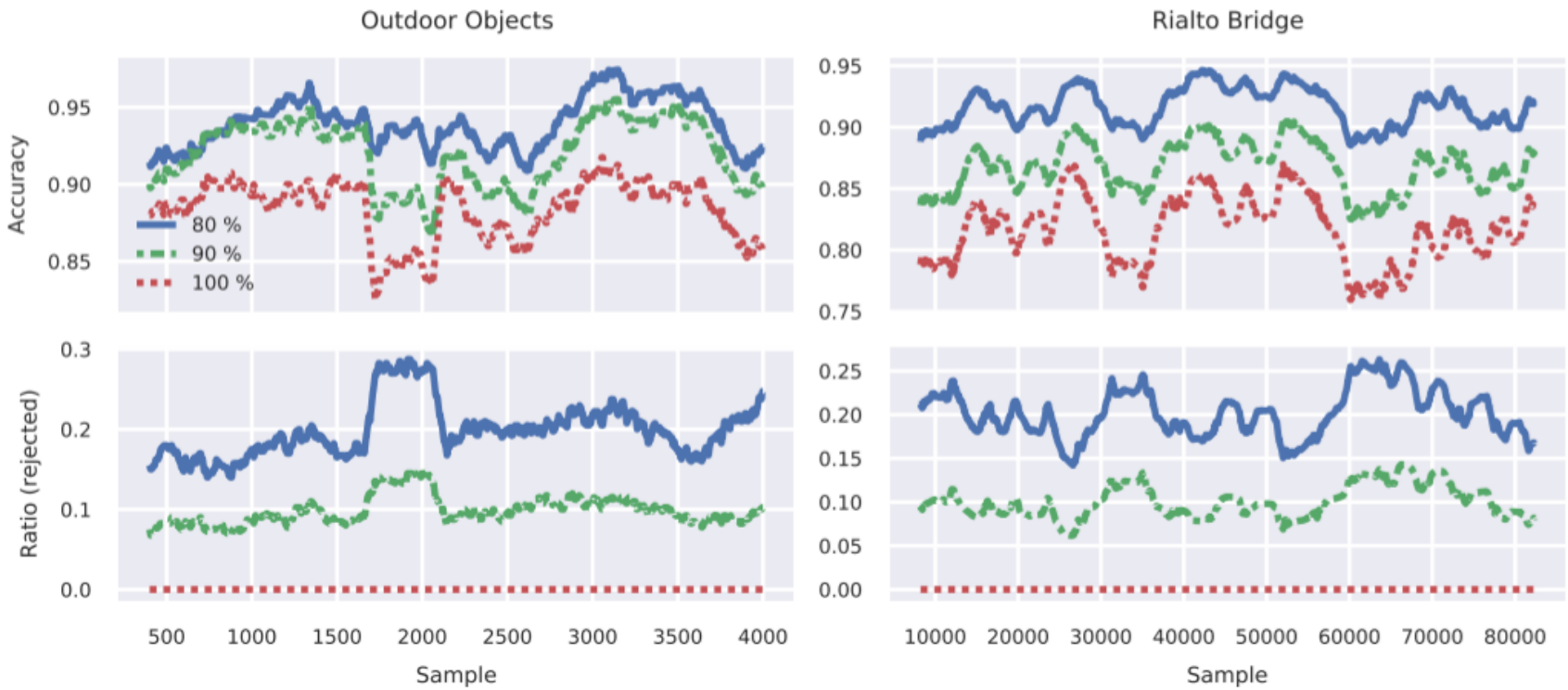
Reject option in data stream classification[45]

1. Estimate $p(y_t \mid \boldsymbol{x}_t, y_{t-1}, ..., t_{t-T})$ with an online learning method
2. Define a threshold on $p(y_t \mid \boldsymbol{x}_t, y_{t-1}, ..., t_{t-T}, \theta)$

[45] Goepfert et al., Mitigating the Adverse Effects of Concept Drift via the Application of Reject Option, ICANN 2018

Reject option in data stream classification[45]

1. Estimate $p(y_t \mid \boldsymbol{x}_t, y_{t-1}, ..., t_{t-T})$ with an online learning method
2. Define a threshold on $p(y_t \mid \boldsymbol{x}_t, y_{t-1}, ..., t_{t-T}, \theta)$



[45] Goepfert et al., Mitigating the Adverse Effects of Concept Drift via the Application of Reject Option, ICANN 2018

Set-valued prediction under dataset shift

- Size control settings: same methods can be used

[46] Cauchois et al., Robust validation: Confident predictions even when distributions shift, Arxiv

[47] Tibshirani et al., Conformal prediction under covariate shift, Neurips 2020

Set-valued prediction under dataset shift

- Size control settings: same methods can be used
- Error control settings: exchangeability no longer holds

$$\begin{aligned} (\boldsymbol{x}_i, y_i) &\sim p_{\text{train}}(\boldsymbol{x}, y) \qquad \forall i = 1, ..., N \quad (i.i.d.) \\ (\boldsymbol{x}_{N+1}, y_{N+1}) &\sim p_{\text{test}}(\boldsymbol{x}, y) \end{aligned}$$

- Conformal prediction cannot be applied

[46] Cauchois et al., Robust validation: Confident predictions even when distributions shift, Arxiv

[47] Tibshirani et al., Conformal prediction under covariate shift, Neurips 2020

Set-valued prediction under dataset shift

- Size control settings: same methods can be used
- Error control settings: exchangeability no longer holds

$$\begin{aligned} (\boldsymbol{x}_i, y_i) &\sim p_{\text{train}}(\boldsymbol{x}, y) \qquad \forall i = 1, ..., N \quad (i.i.d.) \\ (\boldsymbol{x}_{N+1}, y_{N+1}) &\sim p_{\text{test}}(\boldsymbol{x}, y) \end{aligned}$$

- Conformal prediction cannot be applied
- Solution 1: error control by considering f-divergence of non-conformity score distributions from $p_{\text{train}}(\boldsymbol{x}, y)$ to $p_{\text{train}}(\boldsymbol{x}, y)$[46]

[46] Cauchois et al., Robust validation: Confident predictions even when distributions shift, Arxiv

[47] Tibshirani et al., Conformal prediction under covariate shift, Neurips 2020

Set-valued prediction under dataset shift

- Size control settings: same methods can be used
- Error control settings: exchangeability no longer holds

$$\begin{aligned} (\boldsymbol{x}_i, y_i) &\sim p_{\text{train}}(\boldsymbol{x}, y) \qquad \forall i = 1, ..., N \quad (i.i.d.) \\ (\boldsymbol{x}_{N+1}, y_{N+1}) &\sim p_{\text{test}}(\boldsymbol{x}, y) \end{aligned}$$

- Conformal prediction cannot be applied
- Solution 1: error control by considering f-divergence of non-conformity score distributions from $p_{\text{train}}(\boldsymbol{x}, y)$ to $p_{\text{train}}(\boldsymbol{x}, y)$[46]
- Solution 2: error control by considering likelihood ratio between $p_{\text{train}}(\boldsymbol{x})$ and $p_{\text{train}}(\boldsymbol{x})$ (covariate shift only)[47]

[46] Cauchois et al., Robust validation: Confident predictions even when distributions shift, Arxiv

[47] Tibshirani et al., Conformal prediction under covariate shift, Neurips 2020

Conformal prediction under distribution shift[48]

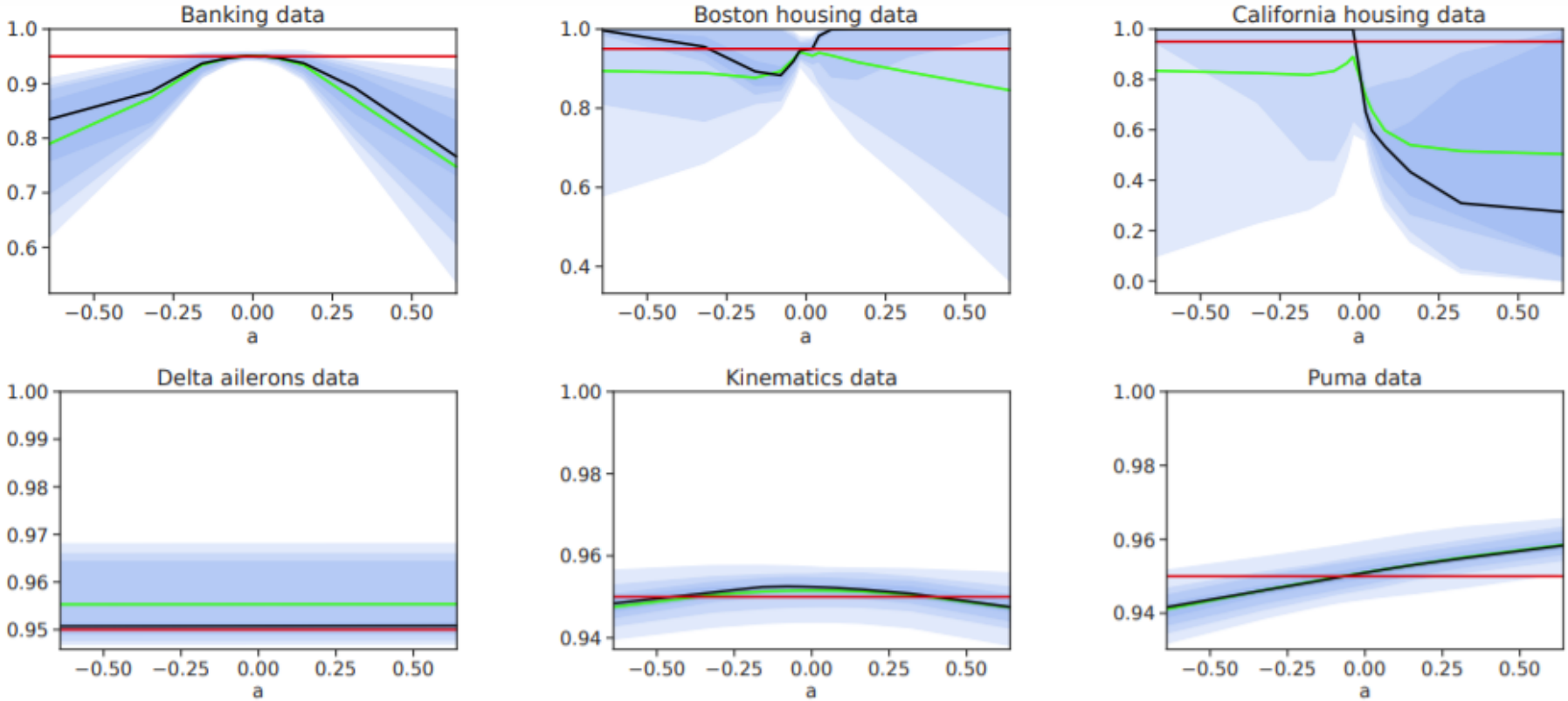


Figure 1. Empirical coverage for the prediction sets generated by the standard conformal methodology across nine regression data sets and 50 random splits of each data set, with an exponential tilting in X space along the first principal component of X. The horizontal axis gives the value of the tilting parameter a; the vertical the coverage level. A green line marks the average coverage, a black line marks the median coverage, and the horizontal red line marks the nominal coverage .95. The blue bands show the coverage at deciles over 50 splits.

[48] Cauchois et al., Robust validation: Confident predictions even when distributions shift, Arxiv

Conformal prediction under distribution shift[49]

- Non-conformity score: $s(\boldsymbol{x}, y) = -f_y(\boldsymbol{x})$

$$\begin{aligned}
(\boldsymbol{x}_i, y_i) &\sim p_{\text{val}}(\boldsymbol{x}, y) \quad \forall i = 1, ..., N \quad (i.i.d.) \\
s(\boldsymbol{x}_i, y_i) &\sim P_{\text{val}} \quad \forall i = 1, ..., N \quad (i.i.d.) \\
(\boldsymbol{X}_{N+1}, Y_{N+1}) &\sim p_{\text{test}}(\boldsymbol{x}, y) \\
s(\boldsymbol{X}_{N+1}, Y_{N+1}) &\sim P_{\text{test}}
\end{aligned}$$

- Required coverage:

$$\mathbb{P}\left(Y_{N+1} \in \Gamma(\boldsymbol{X}_{N+1})\right) \geq 1 - \alpha$$

- f-divergence ball assumption:

$$\text{Quantile}_{WC}(\alpha, P_{\text{val}}) = \sup_{D(P_{\text{test}}, P_{\text{val}}) \leq \rho} \text{Quantile}(\alpha, P_{\text{test}})$$

$$D(P_{\text{test}}, P_{\text{val}}) = \int_{z \in \mathcal{Z}} f\left(\frac{P_{\text{test}}(z)}{P_{\text{val}}(z)}\right) dP_{\text{val}}(z)$$

[49] Cauchois et al., Robust validation: Confident predictions even when distributions shift, Arxiv

Conformal prediction under distribution shift[50]

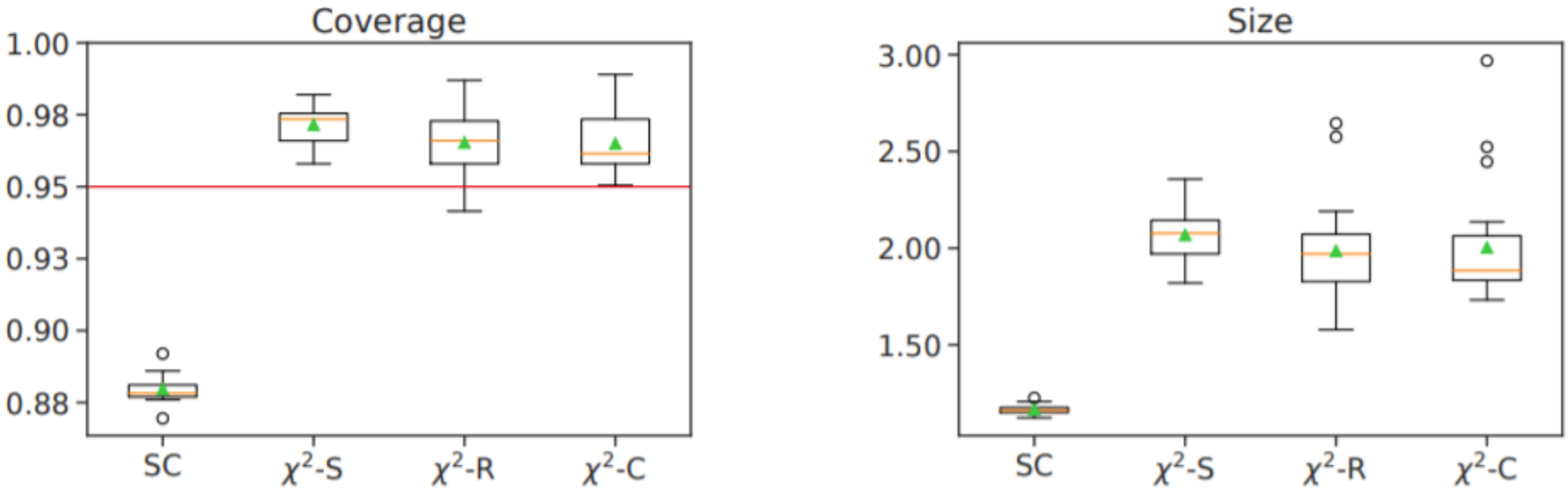


Figure 4. Empirical coverage and average size for the prediction sets generated by the standard conformal methodology ("SC") and the chi-squared divergence, across 20 random splits of the CIFAR-10 data. We set ρ according to the sampling ("χ^2-S"), regression ("χ^2-R"), and classification-based ("χ^2-C") strategies for estimating the amount of shift that we describe in Section in 3. The horizontal red line marks the marginal coverage .95.

[50] Cauchois et al., Robust validation: Confident predictions even when distributions shift, Arxiv

Conclusions

- Important to make a distinction between prior probability shift, covariate shift and concept drift
- The OOD detection literature mainly focuses on **prior probability shift**
- Existing methods can be categorized according to several criteria: amount of supervision, discriminative versus generative models, etc.
- Little work in the **concept drift setting**
- Little work on **set-valued prediction under distribution shift**

Some related events

- ECML/PKDD 2020: Uncertainty in machine learning **(Tutorial+Workshop)**
- ECML/PKDD 2021: Data shift in machine learning: What is it, and what are potential remedies? **(Tutorial)**
- ICML 2021: Distribution-free uncertainty quantification **(Workshop)**
- ICML 2021: Uncertainty and robustness in deep learning **(Workshop)**
- Neurips 2021: Out-of-distribution generalization and adaptation in natural and artificial intelligence **(Workshop)**
- Neurips 2021: Distribution shifts: connecting methods and applications **(Workshop)**