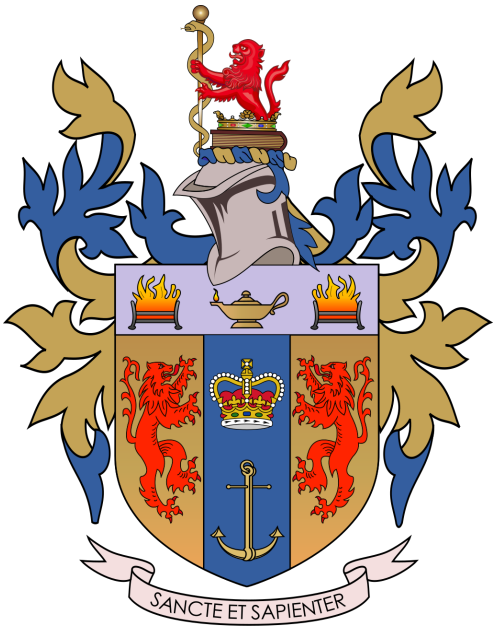
Reliable Machine Learning for Communication Systems: Tools and Applications

Cohen, Kfir

Awarding institution:
King's College London

Reliable Machine Learning for Communication Systems: Tools and Applications



Kfir Maimon Cohen

Department of Engineering
Faculty of Natural, Mathematical & Engineering Sciences

This dissertation is submitted for the degree of
Doctor of Philosophy

King's College London June 2024

Abstract

Two of the main principles underlying the life cycle of an artificial intelligence (AI) module in communication networks are adaptation and monitoring. Adaptation refers to the need to adjust the operation of an AI module depending on the current conditions; while monitoring requires measures of the reliability of an AI module's decisions. Furthermore, the dynamic nature of a communication system imposes short coherent resources due to rapid conditions changes. This makes the available data for module training, e.g., known pilot transmission, to be of small size, requiring the AI module to be sample-efficient.

In the first part of this thesis, we integrate both meta-learning and Bayesian learning for these challenges. Meta-learning addresses sample-efficiency by extracting useful shared properties across different channel conditions by learning how to learn when facing a new condition with few pilots. Bayesian learning increases reliability by producing better-calibrated, e.g., less overconfident, decisions. A well-calibrated AI model is one that can reliably quantify the uncertainty of its decisions, assigning high confidence levels to decisions that are likely to be correct and low confidence levels to decisions that are likely to be erroneous. As an application, we apply the integration of meta-learning and Bayesian learning to symbol demodulation and validate its improvements. The capacity to quantify uncertainty in the model parameter space is further leveraged by extending Bayesian meta-learning to an active setting. In it, the designer can select in a sequential fashion channel conditions under which to generate data for meta-learning from a channel simulator. Bayesian active meta-learning is seen in experiments to significantly reduce the number of frames required to obtain efficient adaptation procedure for new frames of an equalization problem.

While Bayesian meta-learning is better than frequentist learning, it is a best-effort approach with no formal guarantees. To obtain mathematical reliability guarantees, we incorporate in the second part of the thesis the framework of conformal prediction. Conformal prediction post-processes in an ad-hoc manner a probabilistic predictor into set predictor, producing set of labels that is guaranteed to contain the correct label with a probability chosen by the designer. This is done irrespectively of the true, unknown, distribution

underlying the generation of the variables of interest, and can be defined in terms of ensemble or time-averaged probabilities. We apply recent conformal prediction advances, including cross-validation-based schemes that reduce average set sizes, to communication problems such as symbol demodulation and modulation classification. As a communication system may have its data distribution change over time, online conformal prediction is used and investigated for received signal strength prediction, as well for 5G dynamic scheduling of ultra-reliable and low-latency communication traffic. Experiments compare empirical coverage rate and averaged set sizes of different schemes.

Conformal prediction is a special case of conformal risk control for which the risk is the miscoverage indicator. We develop a novel cross-validation-based conformal risk control and show its ability to meet a target risk while keeping the average predictive set sizes smaller than the validation-based counterparts.

In loving memory of my father ***Avi****,*
who instilled in me persistent curiosity and the strive to learn how to learn.

Acknowledgements

Firstly, I would like to thank my family for their unwavering support throughout the research. I could not have asked for a more supportive partner than my wife *Inna*. In good times and less good times, she has always been there for me. Nobody, including myself, has sacrificed more than her. For the advance of my professional career, she has put her impressive career on a pause, even though she was well-appreciated and climbing up the rank ladder. I live with the fact I can never repay her back. My children *Doron*, *Linoy*, and *Romy*, are an endless source of energy for me. Relocation is never easy for a child, and while they were not really given an option, I appreciate the willingness and efforts they made in this adventure. As a family, we feel blessed of the opportunity we are having to be together, and to create memories that will last for a lifetime. I am grateful for my mother *Iris* who always believed in me and provided the means to do what I wanted to. Being remote is hard and I am happy we will be closer in distance soon. I wish to acknowledge my sister *Moran* and my brother *Elad*, for their remote support of my endeavor, and the help and support received by my parents in-law *Larisa* and *Benci*.

Secondly, I cannot express how appreciative I am to be supervised by Professor *Osvaldo Simeone*. Right from the beginning, he kindly embraced me and supplied all necessary conditions for my academic progress, including funding, conducting meaningful discussions, and moral help. By giving me the chance to be his teaching assistant and by sharing preprint drafts of his excellent book, I was able to solidify my understanding of machine learning and communication systems. The first valuable lesson I learned from Osvaldo, is the importance of conveying a message in text through the lens of the reader. I am sure I have much to improve on this matter, yet the critical thinking accompanies my writing. The second valuable lesson is the understanding that taking off excessive text does not fall in importance than writing another piece of text. His devotion is not unique to a specific student, and his ability to lead a group of researchers with personal involvement is admirable.

I would never have delved the depth of the material without my collaborators. A special and warm thanks goes to Dr *Sangwoo Park*, for the fruitful discussions we conducted, and

the notes and coding tricks we have been sharing. I value his help and involvement in each work we had together, and happy we were able to squeeze in socializing beyond professional collaboration. I am grateful for the opportunity to coauthor with Distinguished Professor *Shlomo Shamai* and Professor *Petar Popovski*, who gave thorough reviews and contributed the joint works. Lastly, I value the help of my group member Dr *Nicolas Skatchkovsky* for willing to help whenever I have asked for such, and all group members of the KCLIP lab I have met, past and present. Lastly, I wish to express gratitude to Professor *Iñaki Esnaola* and Dr *Stefan Vlaski* for examining this dissertation, and for great comments that made the thesis more accurate with enhanced readability.

This work has been partially supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement No. 725731, and by King's College London Centre for Doctoral Studies. I am grateful for their financial support, I could not have undertaken this journey without it.

Many events took place in the United Kingdom during the period of my research: starting off with the COVID-19 pandemic outbreak, which led for me to study side by side for months with my kids while they are involuntarily home-schooled; celebration of the 70 years Platinum Jubilee of Queen Elizabeth II; the national grief, months later, of her majesty passing away; and his majesty King Charles III coronation. London has been very welcoming, my family and I will cherish our time here. At the same time, my homeland Israel, where my heart lies, had experienced the global pandemic, political turnovers, and sadly the recent 7 October 2023 murderous attack. I deeply hope for the safe return of all hostages who were kidnapped and are still being kept in hostage with no humanitarian conditions. I wish residents of the land of Israel and its surrounding regions who value life, seek for prosperity, and act in a humane manner – to live peacefully side by side. 🎗

Table of contents

List of figures

List of tables

Nomenclature

Acronyms

K-CV-CP	K-fold Cross-Validation-based Conformal Prediction
K-CV-CRC	K-fold Cross-Validation-based Conformal Risk Control
3GPP	Third Generation Partnership Project
5G	Fifth Generation of mobile communication networks
AI	Artificial Intelligence
AM	Amplitude Modulation
APSK	Amplitude and Phase-Shift Keying
ASK	Amplitude-Shift Keying
BALD	Bayesian Active Learning by Disagreements
BNN	Bayesia Neural Network
CP	Conformal Prediction
CV-CP	Cross-Validation-based Conformal Prediction
CV-CRC	Cross-Validation-based Conformal Risk Control
ECE	Expected Calibration Error
EFC	Exclusive Full-Conformal
eMBB	enhanced Mobile Broadband
ERM	Empirical Risk Minimization

FM	Frequency Modulation
GD	Gradient Descent
GMSK	Gaussian Minimum-Shift Keying
HMC	Hamiltonian Monte Carlo
HVP	Hessian-Vector Product
i.i.d.	Independent and Identically Distributed
I/Q	In-phase and Quadrature
IFC	Inclusive Full-Conformal
IoT	Internet of Things
KL	Kullback-Liebler
LMC	Langevin Monte Carlo
LMMSE	Linear Minimal Mean Square Error
LSTM	Long Short-Term Memory
LTE	Long Term Evolution
MAP	Maximum A Posteriori
MC	Monte Carlo
MIMO	Multiple-Input Multiple-Output
ML	Maximum Likelihood
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NC	Nonconformity
NLOS	Non-Line-of-Sight
NN	Neural Network

NPB	Naïve Probabilistic-Based
NQB	Naïve Quantile-Based
O-RAN	Open Radio Access Network
OOK	On–Off Keying
OQPSK	Offset Quadrature Phase-Shift Keying
PSK	Phase-Shift Keying
QAM	Quadrature Amplitude Modulation
RCI	Rolling Conformal Inference
RSS	Received Signal Strength
SELU	Scaled Exponential Linear Unit
SER	Symbol Error Rate
SGD	Stochastic Gradient Descent
SGLD	Stochastic Gradient Langevin Dynamics
SIMO	Single-Input Multiple-Output
SNR	Signal to Noise Ratio
SVGD	Stein Variational Gradient Descent
URLLC	Ultra-Reliable Low-Latency Communications
VB-CP	Validation-Based Conformal Prediction
VB-CRC	Validation-Based Conformal Risk Control
VI	Variational Inference

Bayesian Learning

$\hat{F}_{\mathcal{D}}(\varphi \mid \xi)$	approximation of variational free energy
$\boldsymbol{\phi}$	random variational parameter vector

$F_{\mathcal{D}}(\varphi \mid \xi)$	variational free energy
$p(\mathcal{D} \mid \phi)$	likelihood of data set given model parameter vector
$p(\phi)$	prior distribution of model parameter
$p(\phi \mid \mathcal{D})$	posterior distribution of model parameter
$q(\phi \mid \mathcal{D})$	approximated Bayesian posterior of $p(\phi \mid \mathcal{D})$
$q(\phi \mid \varphi)$	variational distribution
R	Monte Carlo ensemble size

Symbols associated with Data

$\mathcal{D}$	data set
$\mathcal{D}^{\text{tr}}, \mathcal{D}^{\text{te}}, \mathcal{D}^{\text{val}}$	training/test/validation partitions of a data set
$\mathcal{X}$	covariate space
$\mathcal{Y}$	label space
$N^{\text{tr}}, N^{\text{te}}, N^{\text{val}}$	sizes of training/test/validation data subsets
$\boldsymbol{\mathcal{D}}$	random variable data set
$\mathbf{x}$	covariate random vector
$\mathbf{y}$	label random variable
N	size of data set $\mathcal{D}$
x	covariate vector
y	label
z	covariate-label data pair (x, y)

Active and Meta-Learning

$\mathcal{D}_*^{\text{tr}}, \mathcal{D}_*^{\text{te}}$	training/test data sets for a meta-test frame
$\mathcal{D}_\tau$	data set for the τ-th frame

$\mathcal{D}_\tau^{\text{tr}}, \mathcal{D}_\tau^{\text{te}}$	training/test data sets for the τ-th meta-training frame
$\mathcal{D}_{1:t}$	data sets of all frames indexed $1, \ldots, t$
ϕ_*	adapted model parameter vector for the meta-test frame
ϕ_τ	adapted model parameter vector for the τ-th meta-training frame
ϕ_{t+1}	next model parameter vector after observing t frames
τ	frame index
c_τ	context variable for the τ-th frame
c_{t+1}	requested next context vector after observing t frames
N_τ	size of data set $\mathcal{D}_\tau$ for the τ-th frame
$N_\tau^{\text{tr}}, N_\tau^{\text{te}}$	sizes of training/test data subsets
$s_t(\phi \mid \varphi_{1:t})$	scoring function for a candidate model parameter candidate ϕ

Number sets and symbols

$\emptyset$	empty set
$\jmath$	unit imaginary number $\sqrt{-1}$
$\mathbb{C}$	complex set
$\mathbb{R}$	real set
$\mathbf{1}_d$	d-vector of all ones
I_d	identity $d \times d$ matrix

Operators and Functions

argmax	the arguments of the maxima
argmin	the arguments of the minima
$\circ$	composition operator
$\delta(\cdot)$	Dirac delta function

$\exp(\cdot)$	natural exponential function
$\mathbb{1}(\cdot)$	indicator function
$\mathrm{KL}(\cdot \parallel \cdot)$	Kullback-Liebler divergence
$\log(\cdot)$	natural logarithm function
$\mathrm{Diag}(\cdot)$	diagonal matrix from vector
$\mathrm{erf}(\cdot)$	error function
$\mathrm{H}(p, q)$	cross-entropy between distributions p and q
$\mathrm{sign}(\cdot)$	sign function
$\mathrm{softmax}(\cdot)$	softmax function
∇_ϕ	gradient with respect to ϕ
$\odot$	element-wise multiplication
$K!$	factorial of integer K
$x^\top$	transpose of vector x

Probabilistic Modeling

$\hat{y}(x \mid \mathcal{D})$	data-based hard predictor
ϕ	model parameter vector
$\phi^*_\mathcal{D}$	optimized data-based model parameter vector
φ	variational parameter vector
ξ	model hyperparameter vector
$L_\mathcal{D}(\phi)$	empirical loss
$p(y \mid x, \mathcal{D})$	data-based probabilistic predictor (frequentist or Bayesian)
$p(y \mid x, \phi)$	frequentist probabilistic predictor
$p_\mathcal{D}(x, y)$	empirical distribution following data set $\mathcal{D}$

Probability

Bern	Bernoulli probability mass function
$\mathcal{CN}$	complex normal probability density function
$\mathrm{E}_{\mathbf{x}\sim p(x)}[\cdot]$	expectation over random variable $\mathbf{x}$
$\mathcal{N}$	normal probability density function
$\mathbb{P}(\cdot)$	probability of an event
$p_0(x, y \mid c_\tau)$	ground-truth distribution conditioned on context variable c_τ

Set Predictors

$2^{\mathcal{Y}}$	power set of $\mathcal{Y}$, combination of all subsets including null set
α	desired miscoverage rate
$\ell_q(y, \hat{y})$	pinball loss for quantile q
$\Gamma(x \mid \mathcal{D})$	predictive set of input x using data $\mathcal{D}$
$\mid \mathcal{Y} \mid$	size of set $\mathcal{Y}$
$\mathrm{NC}((x, y) \mid \mathcal{D})$	nonconformity score of point (x, y) to data set $\mathcal{D}$
$\phi_{\mathcal{D},q}$	q-quantile optimized model parameter vector
$Q_\alpha(\cdot)$	empirical α-quantile of a vector

Chapter 1

Introduction

1.1 Motivation

Artificial intelligence (AI) is seen as a key enabler for next-generation wireless systems [3]. Emerging solutions, such as Open-Radio Access Network (O-RAN), incorporate AI modules as native components of a modular architecture that can be fine-tuned to meet the requirements of specific deployments [4]. Two of the main principles underlying the life cycle of an AI module in communication networks are *adaptation* and *monitoring* [5]. Adaptation refers to the need to adjust the operation of an AI module depending on the current conditions, particularly for real-time applications at the frame level. At run time, an AI model should ideally enable monitoring of the quality of its outputs by providing measures of the reliability of its decisions. The availability of such reliability measures is instrumental in supporting several important functionalities, from the combination of multiple models to decisions about retraining [6].

The most common metric to design an AI model and to gauge its performance is the average *accuracy*. However, in applications in which AI decisions are used within a larger system, AI models should not only be as accurate as possible, but they should also be able to reliably quantify the uncertainty of their decisions. As an example, consider an unlicensed link that uses AI tools to predict the best channel to access out of four possible channels. A predictor that assigns the probability vector of $[90\%, 2\%, 5\%, 3\%]$ to the possible channels predicts the same best channel – the first – as a predictor that outputs the probability vector $[30\%, 20\%, 25\%, 25\%]$. However, the latter predictor is less certain of its decision, and it may be preferable for the unlicensed link to refrain from accessing the channel when acting on less confident predictions, e.g., to avoid excessive interference to licensed links [7, 8].

As in the example above, AI models typically report a confidence measure associated with each prediction, which reflects the model's *self-evaluation* of the accuracy of a decision. Notably, neural network models implement *probabilistic predictors* that produce a probability distribution across all possible values of the output variable. The self-reported model confidence, however, may not be a reliable measure of the true, unknown, accuracy of a prediction. In such situations, the AI model is said to be poorly calibrated.

Classical frequentist learning methods for the design of AI modules fall short on both counts of adaptation and monitoring (see, e.g., [9, 10]). First, conventional frequentist learning is well known to provide inaccurate measures of reliability, typically producing overconfident decisions [10]. Second, the standard learning approach prescribes the one-off optimization of an AI model, hence failing to capture the need for adaptation. We start by investigating the integration of meta-learning and Bayesian learning as a means to address both challenges. As we detail later, Bayesian learning can provide well-calibrated, and hence reliable, measures of uncertainty of a model's decision; while meta-learning can reduce the amount of data required for adaptation to a new task, thus improving efficiency. As a specific use case, we focus on the problems of demodulation and equalization over a fading channel based on the availability of few pilots (see Fig. 1.3). The goal is to develop AI solutions that are capable of adapting the demodulator/equalizer to changing conditions based on few training symbols, while also being able to quantify the uncertainty of the AI model's output.

Deep learning models tend to produce either overconfident decisions [10], or calibration levels that rely on strong assumptions about the ground-truth, unknown, data generation mechanism [11–16]. This dissertation further investigates the use of *conformal prediction* (CP) [17–19] as a framework to design provably well-calibrated AI predictors, with *distribution-free* calibration guarantees that do not require making any assumption about the ground-truth data generation mechanism.

1.2 Research Questions

This research addresses the following questions

- **Research Question 1: Is it possible to enhance the reliability of learning-based communication algorithms in the challenging regime of few pilot symbols?**

Chapter 3 reviews Bayesian learning and shows it can quantify its uncertainty better than conventional learning. Moreover, we explore the integration of Bayesian learning and

meta-learning, to show how knowledge acquired over past communication frames can be utilized for a new frame, in a way that requires few pilots for symbol demodulation.

- **Research Question 2: Can an active selection of training conditions speed up training for new communication settings?**

In Chapter 3 we show that by actively selecting the conditions governing how data is generated, an equalizer can be trained faster by requiring less channel uses than the case when conditions are drawn at random.

- **Research Question 3: Is it possible to provide formal guarantees of reliability for learning-based communication protocols?**

The answer to this question is yes. In Chapter 4 we adopt predictions that are sets of labels, and employ several schemes with the goal of reducing the predictive set sizes. By using conformal prediction, formal guarantees are achieved. In Chapter 5 we apply this to a 5G dynamic services scheduling, and show that both the stringent ultra-reliability and the low-latency constraints are met. Furthermore, in Chapter 6 we develop a more efficient setting, which keeps a formal guarantee while bounding an arbitrary risk.

1.3 Overview

In this work, we are interested in tools that increase the reliability of AI predictors. The thesis can coarsely be divided into two frameworks:

1. The first framework is *probabilistic predictors*, as depicted in the top panel of Fig. 1.1, which is upgraded using three mathematical tools:

   (a) **Bayesian learning**, that treats the network parameters as random variables drawn according to a learnable distribution, rather than scalar values.

   (b) **Meta-learning** that leverages experience acquired over related learning tasks, requiring few labeled samples from the unseen test task before prediction.

   (c) **Bayesian active meta-learning** that actively selects which learning task's data should be available for the learning model next, aiming to learn as fast as possible using few tasks in total.

   We integrate all of these tools and produce an Bayesian active meta-learning which is able to learn how to learn, in a way that reflects uncertainties it has regarding its

predictions, and does so using small amount of labeled test data. This integration is depicted in Fig. 1.2. This is studied in **Chapter 3**.

2. The second framework is *conformal prediction*, which builds on top of probabilistic predictors as a calibration post-process, in a way that formal guarantees about the loss are achieved. The bottom panel of Fig. 1.1 shows that set predictors produce a set of labels rather than a probability vector, and shows the two important prediction metrics characterizing them: coverage and efficiency. This is studied in **Chapters 4 - 5**, and is extended in **Chapter 6**.

1.4 Literature Survey

In this section, we review the various mathematical frameworks that are pillar to answer the research questions listed in Section 1.2, as well as closely related fields

1.4.1 Mathematical Tools Directly Used

Most work on AI for communications relies on conventional frequentist learning tools (see, e.g., the review papers [20–23]). **Frequentist learning** is based on the minimization of the (regularized) training loss, which is interpreted as an estimate of the ground-truth population loss. Frequentist learning assigns a single value to each model parameter as a result of training. This neglects (epistemic) uncertainty that exists at the level of model parameters due to the limited availability of data. When data is scarce, this estimate is unreliable, and hence the focus on a single, optimized, model parameter vector often yields probabilistic predictors that are poorly calibrated, producing overconfident decisions [10, 24–26].

Bayesian learning offers a principled way to address this problem [27, 28]. This is done by producing as the output of the learning process not a single model parameter vector, but rather a distribution in the model parameter space, which quantifies the model's epistemic uncertainty caused by limited access to data. A model trained via Bayesian learning produces probabilistic predictions that are averaged over the trained model parameter distribution. This *ensembling* approach to prediction ensures that disagreements among models that fit the training data (almost) equally well are accounted for, substantially improving model calibration [29, 30]. Bayesian learning can express uncertainty about the true value of the model parameter vector by optimizing over a distribution, rather than over a single point value [31]. By averaging predictions over the distribution of the model

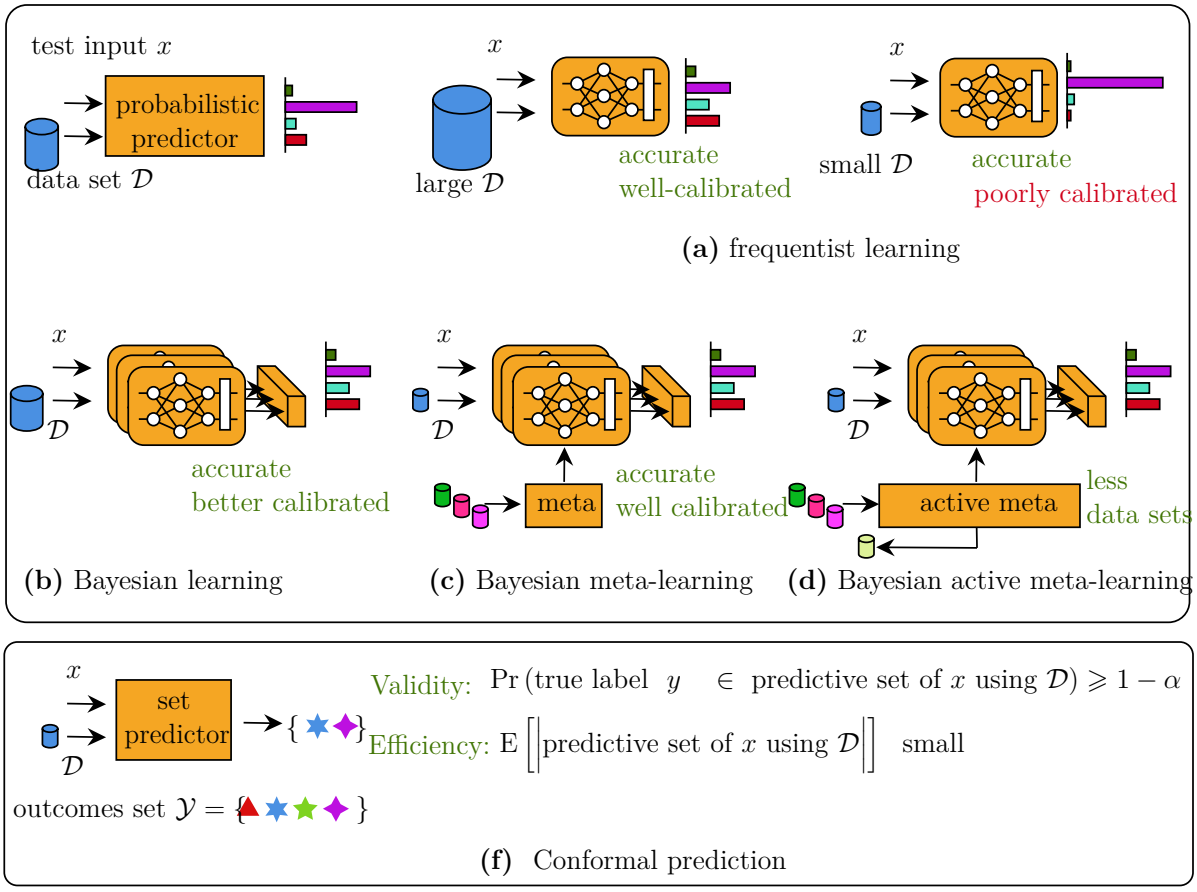


Fig. 1.1 Thesis overview. **(top)** probabilistic predictor. **(a)** frequentist learning is known to be accurate and well-calibrated for large data. For limited availability of data, they tend to produce low quality of uncertainty quantification. **(b)** Bayesian learning learns an ensemble of models, and in general better calibrated. **(c)** Bayesian meta-learning enables to adapt using very small data set, by transferring knowledge acquired over related tasks. **(d)** Bayesian active meta-learning selects the next data set, converging with less data sets. **(bottom)** Conformal prediction uses set predictions. **(f)** formal guarantees on validity, while aiming for efficiency reflected by small predictive set sizes.

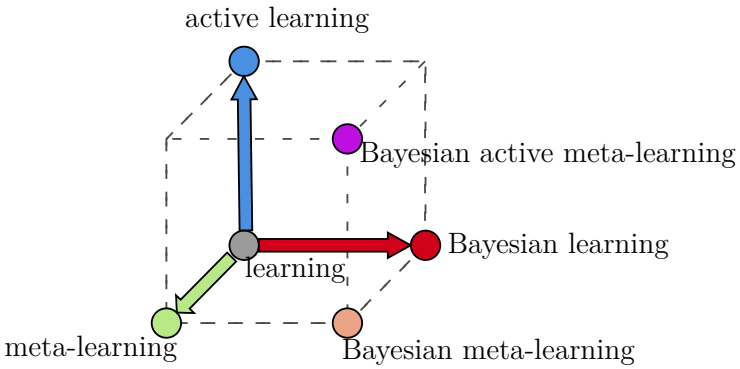


Fig. 1.2 Illustration of the three frameworks encompassing Bayesian active meta-learning. Originating from the base "learning", three frameworks are discussed in this paper: **(red)** Bayesian learning; **(green)** meta-learning; and **(blue)** active learning. Integration of all three yields "Bayesian active meta-learning".

parameters, Bayesian learning is known to be capable of providing decisions that are well calibrated [29, 32, 33]. Calibration refers to the capacity of a model to produce confidence levels that reproduce well the actual accuracy of the decisions.

Bayesian learning can be implemented either via parametric or non-parametric approximations. Adopting a parametric formulation offers computational advantages, scalability, flexibility, interpretability, and enables efficient optimization techniques. On the one hand, **parametric variational inference** (VI), which approximate the exact Bayesian posterior distribution with a tractable variational density [29, 34–37], yields simple and tractable solutions which can scale to large data sets and complex models, while still retaining expressiveness and flexibility of the solution. A key advantage is the efficient optimization of a well defined optimization problem. On the other hand, **non-parametric variational inference** methods do not impose any restriction of the variational distribution to be a member of a predefined parametric family. These methods offer greater flexibility, capture more complex dependencies of the data, and may improve generalization. A widely used non-parametric VI method is the Stein variational gradient descent (SVGD) [38], which optimize over deterministic and interacting particles. For Monte Carlo (MC) techniques, solutions range from first-order Langevin dynamics techniques [39] to more complex methods such as cyclical stochastic gradient Markov chain Monte Carlo [40] and Hamiltonian Monte Carlo (HMC) [41]. Implementing any of these schemes for a specific engineering application is a non-trivial task. Each class of methods comes with its own set of technical challenges and engineering choices. For instance, VI requires the selection of a variational distribution family, such as mean-field Gaussian models, and the specification of a stochastic optimization algorithm. The limitation of the parametric VI lies in the low expressiveness

levels of parametric distributions, which is an inherent property by limiting to a family of structured distributions. This requires to strike a balance between model expressiveness and computational efficiency when choosing the parametric variational family to be versatile enough. Non-parametric VI may turn to be computational complex and it scales poorly to high dimensions or large data. In this dissertation, the parametric VI is chosen over non-parametric VI due to its practically and applicability.

Meta-learning, also known as *learning to learn*, optimizes training strategies that can fine-tune a model based on few samples for a new task by transferring knowledge across different learning tasks [42–48]. Meta-learning is a natural tool to produce AI solutions that are optimized for adaptation. Prior work on meta-learning for communication systems, including [49–58], is limited to standard frequentist learning. Therefore, existing art is unable to produce models that provide well-calibrated estimations of reliability. Most related to our work is [49], which proposes to leverage pilot information from previous frames in order to optimize training procedures to be applied to the pilots of new frames (see Fig. 1.3).

Bayesian meta-learning aims at optimizing the procedure that produces the posterior distribution for new learning tasks. Accordingly, the goal of Bayesian meta-learning is to enhance the efficiency of Bayesian learning by reducing the number of training points needed to obtain accurate and well calibrated Bayesian models. The optimization of the Bayesian learning process is carried out by transferring knowledge from previously encountered tasks for which data are assumed to be available [59, 60, 24].

Beside meta-learning, another approach to reduce the number of required training data points is **active learning** [61–65]. Active learning amounts to the process of choosing which samples should be annotated next and incrementally added to the training set [66]. Through this process, active learning can select relevant samples at which the model is currently most uncertain in order to speed up the training process.

A much less studied area is **active meta-learning**, which aims at reducing the number of tasks a meta-learner must collect data from, before it can adapt efficiently to new tasks [65, 67]. Reference [65] proposes a method based on Bayesian meta-learning via empirical Bayes; while the paper [67] takes a hierarchical Bayesian approach, generalizing the Bayesian active learning by disagreements (BALD) criterion introduced in [61] to meta-learning. While [65] assumes labeled training sets, reference [67] considers unlabeled data during active meta-learning. As such, the setting in it is not applicable to the problem under study here in which data consists of supervised pairs of pilots and received signals (see Fig. 1.3). A summary of the relevant approaches built upon in this work is given in Table 1.1.

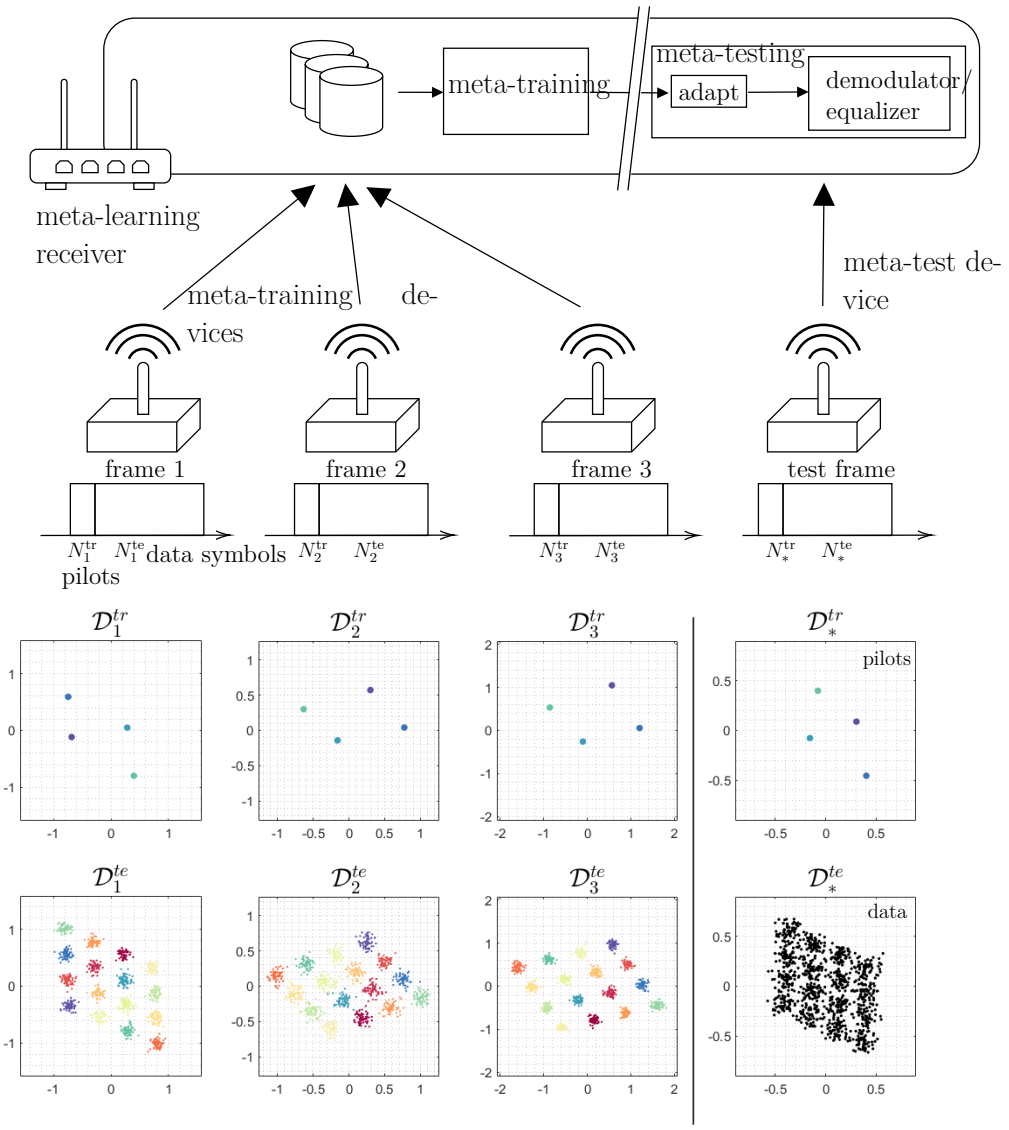


Fig. 1.3 Illustration of the meta-learning problem studied in this work for the example of 16-ary quadrature amplitude modulation (16-QAM). A receiver has available data corresponding to frames previously received from multiple devices, each possibly experiencing different channel conditions. Given meta-training data sets $\{\mathcal{D}_\tau\}_{\tau=1}^{t}$ of pilots from previous frames, partitioned into training data and test data, the demodulator optimizes a hyperparameter vector ξ. For a newly received frame, the receiver uses the few pilots therein to adapt the demodulator/equalizer parameter vector ϕ_*. In the Bayesian meta-learning framework, instead of a single parameter vector ϕ_*, the receiver optimizes over an ensemble of parameter vectors through the hyperparameter vector ξ of a posterior distribution $p(\phi_*|\mathcal{D}_*^{\text{tr}}, \xi)$.

Table 1.1 A Summary of the relevant techniques considered in this work

Approach	Goal	Methodology
frequentist learning	accurate data-driven predictions	minimize the training loss over the model parameter vector ϕ
Bayesian learning via variational inference	reliable and accurate data-driven predictions	minimize the free energy over the parameters φ of the variational distribution $q(\phi\|\varphi)$
frequentist meta-learning	sample efficiency	minimize the meta-training loss over hyperparameters ξ to be used for frequentist learning
Bayesian meta-learning	sample efficiency and reliability	minimize the meta-training loss over hyperparameters ξ to be used for Bayesian learning
Bayesian active meta-learning	task efficiency, sample efficiency, and reliability	minimize the meta-training loss over the hyperparameters ξ and over the sequential selection of meta-learning tasks
conformal prediction	coverage guarantees	produce a predictive set of labels, and regulate its size via a validation set
cross-validation conformal prediction	efficient conformal prediction	produce efficient predictive sets via folds
online conformal prediction	coverage guarantees	regulate in an online fashion a calibration parameter to produce smallest set to meet miscoverage
conformal risk control	risk guarantees	produce a predictive set of labels while bounding arbitrary risk
cross-validation conformal risk control	efficient conformal risk controlling	produce efficient predictive sets with bound arbitrary risk via folds

Exact Bayesian learning offers formal guarantees of calibration only under the assumption that the assumed model is *well specified* [12, 11]. In practice, this means that the assumed neural network models should have sufficient capacity to represent the ground-truth data generation mechanism, and that the predictive uncertainty should be unimodal for continuous outputs (since conventional likelihoods are unimodal, e.g., Gaussian) [12, 29, 28]. These assumptions are easily violated in practice, especially in communication systems in which lower-complexity models must be implemented on edge devices, and access to data for specific network configurations is limited. Specific examples are provided in [26] for applications including modulation classification [68, 69] and localization [70, 71].

Robustified versions of Bayesian learning that are based on the optimization of a modified free energy criterion were shown empirically to partly address the problem of model misspecification [11, 12], with implications for communication systems presented in [26]. However, robust Bayesian learning solutions do not have formal guarantees of calibration in the presence of misspecified models.

Another family of methods that aim at enhancing the calibration of probabilistic models implement a validation-based post-processing phase. Platt scaling [72] and temperature scaling [10] find a fixed parametric mapping of the trained model output that minimizes the validation loss, while isotonic regression [73] applies a non-parametric binning approach. These recalibration-based approaches cannot guarantee calibration, as they may overfit the validation data set [74] and they are sensitive to the inaccuracy of the starting model [75].

Conformal prediction is a general framework for the design of set predictors that satisfy formal, *distribution-free*, guarantees of calibration [17, 18]. Given a desired miscoverage probability α, CP returns set predictions that include the correct output value with probability at least $1-\alpha$ under the only assumption that the data distribution is *exchangeable*. A sequence of random variables is considered exchangeable if the order in which the random variables are observed does not affect their joint distribution. This condition is less stringent than the standard assumption of "i.i.d." data made in the design of most machine learning systems, with the latter being a special case of the former.

The original work on CP, [17], introduced **validation-based CP (VB-CP)** and *full CP*. Since then, progress has been made on reducing computational complexity, minimizing the size of the prediction sets, and further alleviating the assumptions of exchangeability. The authors of [76, 77] proposed the optimization of a *CP-aware loss* to improve the efficiency of validation-based CP, while avoiding the larger computational cost of cross-validation. The work [78] proposed reweighting as a means to handle distribution shifts between the examples in the data set and the test point. Other research directions include improvements in the training algorithms [79, 80], and the introduction of novel calibration metrics [81, 82].

Finally, **online CP**, presented in [83, 84], was shown to achieve long-term calibration over time without requiring statistical assumptions on the data generation.

CP provably guarantees calibration at the cost of relaxing point predictions to sets of point predictions. Another line of work on formal, distribution-free, post-hoc calibration introduces *Venn predictors* which relax probabilistic predictors to sets of probabilistic predictors [17, Sec. 6], [85–87]. Such sets contain at least one well-calibrated probabilistic predictor, and the size of each set is no larger than the number of possible labels. In general, the sets of probabilistic predictor produced by Venn predictors, unlike those output by CP, do not appear to offer actionable information for decision processes. In contrast, Venn predictors have been explored in [88] for *explanation* purposes via perturbations of input features [89].

Cross-validation-based CP (CV-CP) was proposed in [90] to reduce the computational complexity as compared to full CP, while improving the efficiency of validation-based CP. This is done by partitioning the available data set into folds, each acting as a calibration set of the leave-fold-out data, and as such requires more models to be trained than VB-CP.

Conformal risk control (CRC) [91] generalized the CP framework to arbitrary risks, beyond the miscoverage loss. By introducing a threshold parameter that controls the predictive set size, it was proved that CP is a private case of CRC. Most importantly, the predictive set is built upon a black-box predictor as a post-processing procedure. This line of research includes also distribution-free CRC [92], online CRC [84], diffusion models [93], and calibration algorithms [94].

1.4.2 Closely Related Fields

A useful and practical statistical tool for uncertainty quantification when the population distribution is unknown is **bootstrapping**. Provided generated data of limited size, bootstrapping may be used to estimate distributional properties, like confidence intervals or mean. Three steps are performed repeatedly in it. The first step is resampling with replacement, whereby each data point is selected at random with replacement from the original data set, producing a resampled data set with some points occurring more than once, while others are absent. The second step involves computation of a desired statistic for each of the resampled sets. The third, last, step includes computing the distribution of the bootstrapped statistics. This allows empirical estimation without any strong assumptions. A meta-learning scheme that leverages bootstrapping is in [95].

Transfer learning supports fast adaptation of systems to new tasks. It involves two related learning tasks: the source task; and the target task. Typically, the available data

corresponding the target task is too small to allow proper generalization by relying solely on it while learning. The main idea of transfer learning is that knowledge extracted from source task generated data is useful, hence transferable, to enhance the generalization performance of the target task. In order to reuse experience gained over the source task in the target task, the two tasks should be related to some extend, ranging from covariate shift for which only the covariate distribution shift between the two tasks while the distribution of the label conditioned on the covariate stays the same, to the less restrictive general shift, where there is no discriminative common model. Transfer learning is useful in various real-life domains, including computer vision [96], natural language processing [97], and speech recognition [98]. Surveys of Transfer learning are in [99] and [28]. Most approaches of transfer learning include a fine-tuning of the transferred parameters which were trained over the source task data, using a training objective associated with the source task. Hence, adaptation to the target task is done independently after extracting information of the source task. In contrast, meta-learning considers a meta-objective while it meta-trains over the meta-training. This meta-objective governs the learning to adapt to a new task.

Meta-learning methods can be categorized into three main approaches. The first approach, **optimization-based meta-learning**, solves via optimization a task-specific problem, and extracts meta-knowledge in a way that helps the optimization across the meta-tasks. The meta-knowledge is acquired by optimizing a meta-objective that guides the meta-learning process. Predominant examples for this are the MAML [43], which learn an initialization of model parameters that facilitates rapid adaptation to new tasks or datasets through optimization processes, and REPTILE [100]. These algorithms usually are more computational complex than the other approaches due to second-order optimization, with first-order approximations reported in [43]. The second approach is **model-based meta-learning**, also known as **black box meta-learning**. In it, a flexible model-agnostic generic representation that captures similarities between tasks is learned. Efficient generalization is done by mapping data points to task-specific representations. Methods of this approach support different model architectures across tasks and are easily combined with learning problems like supervised learning reinforcement learning. In this approach, the lack of inductive bias makes the optimization non-trivial and data-inefficient. Examples for this approach were reported using recurrent networks in [101] and by prediction parameters from activations in [102]. The third approach is **metric-based meta-learning**, also known as **non-parametric meta-learning**. This expressive and computational-fast approach focuses on learning a metric space where task similarities can be measured, enabling efficient generalization across tasks by comparing validation points with training points. Predominant examples of the the metric-based meta-learning are Siamese Networks [103]

and Matching Networks [104]. A survey of these three categories, as well as other related topics, can be found in [105].

An intermediate solution between the two extremes of physics-based modeling and pure data-driven machine learning, is **physics-informed machine learning**. It leverages physical principles and information known about the problem to guide a learning process by regularizing or constraining. This integration of physical knowledge penalizes solutions that do not adhere to physical models, and may increase model prediction and explainability of the solution. Physics-informed machine learning is able to solve complex problems by unlocking the powerful capabilities of machine learning of capturing patterns, with guidance and regularization by the physics model. More ways to enrich models' understanding are **expert knowledge embedding techniques**. These methods incorporate expert knowledge or domain-specific information into machine learning models, typically through embedding representations. Some examples include: manually crafting features which are later used in machine learning models; wrapping a learned model with logic such that the total behavior comply to some constraints; and expert-guided embedding learning techniques that leverage expert-provided supervision to guide the learning process [106]. Expert knowledge embedding techniques leverage domain-specific insights and aim to enhanced robustness and reliability of predictions in real-world problems.

1.5 Contributions

Chapter 3 introduces the use of Bayesian meta-learning to enable both adaptation and monitoring for the tasks of demodulation and equalization. Unlike prior works that considered either frequentist meta-learning [49, 9, 50–53, 56, 54, 55] or Bayesian learning [107–110], the proposed Bayesian meta-learning methodology enables both resource-efficient adaptation and a reliable quantification of uncertainty. To further improve the efficiency of Bayesian meta-learning we propose the use of active meta-learning, which reduces the number of required meta-training data from previously received frames. Specific contributions are as follows.

- We introduce Bayesian meta-learning for the problems of demodulation and equalization from few pilots. The proposed implementation is derived based on parametric VI.
- We introduce Bayesian active meta-learning as a solution to reduce the number of frames required by meta-learning. Active meta-learning selects in a sequential fashion

channel conditions under which to generate data for meta-learning from a channel simulator.

- Extensive experimental results demonstrate that Bayesian meta-learning produces demodulators and equalizers that offer better calibrated soft decisions. Furthermore, they show that for a target meta-testing loss, active meta-learning can reduce the number of simulated meta-training frames required.

Chapter 4 is built to be self-contained, presenting CP from first principles and including both offline and online CP schemes. We investigate applications of CP to symbol demodulation, modulation classification and to channel prediction by leveraging real-world data sets. To the best of our knowledge, our works [111, 112] are the first to investigate the application of CP to the design of AI models for communication systems. The main contributions of this chapter are as follows.

- We provide a self-contained introduction to CP by focusing on validation-based CP [17], cross-validation-based CP [90], and online conformal prediction [84]. The presentation details connections to conventional probabilistic predictors, as well as the performance metrics used to assess calibration and efficiency.

- We propose the application of offline CP to the problems of symbol demodulation and modulation classification. The experimental results validate the theoretical property of CP methods of providing well-calibrated decisions. Furthermore, they demonstrate that naïve predictors that only rely on the output of either frequentist or Bayesian learning tools often result in poor calibration.

- Finally, we study the application of online CP to the problem of predicting received signal strength for over-the-air measured signals [1, 2]. We demonstrate that online CP can obtain the predefined target long-term coverage rate at the cost of negligible increase in the prediction interval as compared to naïve predictors.

In **Chapter 5**, we propose for the first time the application of CP as a design methodology to ensure reliability requirements that hold irrespectively of any modeling or data availability assumptions. Specifically,

- We introduce a CP-based resource allocation scheme for URLLC traffic that makes use of *any* existing model-based or data-driven predictor, offering theoretical reliability guarantees that apply even when the predictor is poorly designed, e.g., due to limited availability of data.

- The proposed CP-based scheduler is shown via experiments to be capable of efficiently adapting to URLLC traffic, providing eMBB users with a larger fraction of spectral resources as compared to conventional schedulers.

In **Chapter 6**, we contribute by integrating for the first time advances made on conformal risk control (CRC) [91] with the cross-validation (CV) paradigm that was reported for CP.

- We propose a new design, that leverages CV into the control of the threshold for the conformal risk control that is used for losses beyond the miscoverage loss.
- By partitioning available data into folds, we numerically show a setting for which this proposal ends with more efficient set predictor for the small amount of data regime.

1.5.1 Reproducibility

For reproducibility purposes, we have made all of our code publicly available, as in Table 1.2. Note that the code of Chapter 3 is aligned to [25] which uses the information theory notations: x is the channel input, and also the predictor's target; while y is the channel output, and hence the predictor covariate. In this dissertation, we use machine learning notations, to be aligned with the consecutive chapters on conformal prediction.

Table 1.2 Available code

	description	repository
Chapter 3	Bayesian active meta - learning	`github.com/kclip/bayesian_active_meta_learning`
Chapter 4	conformal prediction	`github.com/kclip/cp4wireless`
Chapter 5	online CP for URLLC	`github.com/kclip/online_cp_urllc`
Chapter 6	cross-validation CRC	`github.com/kclip/cvcrc`

1.6 Publications

1.6.1 Papers Included in this Manuscript

Journal Papers

- [25] **Kfir M. Cohen**, Sangwoo Park, Osvaldo Simeone, and Shlomo Shamai, "Bayesian Active Meta-Learning for Reliable and Efficient AI-Based Demodulation," *IEEE Transactions on Signal Processing*, vol. 70, pp. 5366–5380, November 2022.

- [111] **Kfir M. Cohen**, Sangwoo Park, Osvaldo Simeone, Petar Popovski, and Shlomo Shamai, "Guaranteed Dynamic Scheduling of Ultra-Reliable Low-Latency Traffic via Conformal Prediction," *IEEE Signal Processing Letters*, vol. 30, pp. 473-477, April 2023.

- [113] **Kfir M. Cohen**, Sangwoo Park, Osvaldo Simeone, and Shlomo Shamai, "Calibrating AI Models for Wireless Communications via Conformal Prediction," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 1, pp. 296-312, September 2023.

Conference Papers

- [114] **Kfir M. Cohen**, Sangwoo Park, Osvaldo Simeone, and Shlomo Shamai, "Learning to Learn to Demodulate with Uncertainty Quantification via Bayesian Meta-Learning," in *Proceedings of 25th International ITG Workshop on Smart Antennas in EURECOM, France (WSA2021)*, pp 202-207, November 2021.

- [112] **Kfir M. Cohen**, Sangwoo Park, Osvaldo Simeone, and Shlomo Shamai, "Calibrating AI Models for Few-Shot Demodulation via Conformal Prediction," in *2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2023)*, pp. 1–5, June 2023.

- [115] **Kfir M. Cohen**, Sangwoo Park, Osvaldo Simeone, and Shlomo Shamai, "Cross-Validation Conformal Risk Control," accepted to *IEEE International Symposium on Information Theory Proceedings (ISIT2024)*, July 2024.

1.6.2 Papers Not Included in this Manuscript

Journal Papers

- [116] Sangwoo Park, **Kfir M. Cohen**, and Osvaldo Simeone, "Few-Shot Calibration of Set Predictors via Meta-Learned Cross-Validation-Based Conformal Prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 01, pp. 280-291, January 2024.

Conference Papers

- [117] Sangwoo Park, **Kfir M. Cohen**, and Osvaldo Simeone, "Few-shot Calibration of Set Predictors via Meta-Learned Cross-Validation-Based Conformal Prediction," in *Sixth Workshop on Meta-Learning at the Conference on Neural Information Processing Systems (NeurIPS2022)*, November 2022.

Chapter 2

Background

In this chapter, we detail the mathematical tools listed in Chapter 1.4.1. The basics covered here, as well as the performance metrics reviewed, are used as a background for the chapters to follow.

2.1 Probabilistic Predictors

Probabilistic predictors implement a parametric conditional distribution model $p(y|x,\phi)$ on the output $y \in \mathcal{Y}$ given the input $x \in \mathcal{X}$, where $\phi \in \Phi$ is a vector of model parameters. Given the training data set $\mathcal{D}$, frequentist learning produces an optimized single vector $\phi^*_{\mathcal{D}}$, while Bayesian learning returns a distribution $q^*(\phi|\mathcal{D})$ on the model parameter space Φ [29, 28]. In either case, we will denote as $p(y|x,\mathcal{D})$ the resulting optimized predictive distribution

$$p(y|x,\mathcal{D}) = \begin{cases} p(y|x,\phi^*_{\mathcal{D}}) & \text{for frequentist learning} \\ \mathrm{E}_{\boldsymbol{\phi}\sim q^*(\phi|\mathcal{D})}[p(y|x,\boldsymbol{\phi})] & \text{for Bayesian learning.} \end{cases} \tag{2.1}$$

Note that the predictive distribution for Bayesian learning is obtained by averaging, or ensembling, over the optimized distribution $q^*(\phi|\mathcal{D})$.

From (2.1), one can obtain a point prediction $\hat{y}$ for output y given input x as the probability-maximizing output as

$$\hat{y}(x|\mathcal{D}) = \operatorname*{argmax}_{y'\in\mathcal{Y}} p(y'|x,\mathcal{D}). \tag{2.2}$$

In the case of a discrete set $\mathcal{Y}$, the hard predictor (2.2) minimizes the probability of detection error under the model $p(y|x, \mathcal{D})$. The probabilistic prediction $p(y|x, \mathcal{D})$ also provides a measure of predictive uncertainty for all possible outputs $y \in \mathcal{Y}$. In particular, for the point prediction $\hat{y}(x|\mathcal{D})$ in (2.2), we have the predictive, self-reported, *confidence level*

$$\text{conf}(x|\mathcal{D}) = \max_{y' \in \mathcal{Y}} p(y'|x, \mathcal{D}) = p\Big(\hat{y}(x|\mathcal{D})\Big|x, \mathcal{D}\Big). \tag{2.3}$$

2.2 Properties of Probabilistic Predictors

As illustrated in Fig. 2.1, the performance of a probabilistic predictor can be evaluated in terms of both accuracy and *calibration*, with the latter quantifying the quality of uncertainty quantification via the confidence level (2.3) [10].

Specifically, a probabilistic predictor $p(y|x, \mathcal{D})$ is said to be *accurate* if its hard predictors $\hat{y}(x|\mathcal{D})$ (2.2) equal the true label y with high probability. The more likely for it to happen, the more accurate the predictor is.

A probabilistic predictor $p(y|x, \mathcal{D})$ is said to be *well calibrated* [10] if the probability that the hard predictor $\hat{y} = \hat{y}(x|\mathcal{D})$ equals the true label matches its confidence level π for all possible values of probability $\pi \in [0, 1]$. Mathematically, calibration is defined by the condition

$$\mathbb{P}\Big(\mathbf{y} = \hat{y}\Big|p(\hat{y}|\mathbf{x}, \mathcal{D}) = \pi\Big) = \pi, \qquad \text{for all } \pi \in [0, 1] \tag{2.4}$$

where the probability $\mathbb{P}(\cdot)$ follows the ground-truth distribution $p_0(x, y)$. Stronger definitions, like that introduced in [118], require the predictive distribution to match the ground-truth distribution also for values of y that are distinct from (2.2).

The most common metric used to quantify the performance of a probabilistic predictor is its *accuracy*, which is measured by the probability that the hard decision $\hat{y}(x|\mathcal{D})$ in (2.2) coincides with the true label y on test data

$$\text{acc}(\mathcal{D}) = \mathbb{P}\Big(\mathbf{y} = \hat{y}(\mathbf{x}|\mathcal{D})\Big) \tag{2.5}$$

with the random variables following the population distribution $(\mathbf{x}, \mathbf{y}) \sim p_0(x, y)$. The accuracy, however, does not capture the quality of the confidence level (2.3) in terms of uncertainty quantification. This aspect is instead quantified by means of calibration metrics. A predictor is *well-calibrated* if its reported confidence levels (2.3) reflect the ground-truth probability of correct detection for each input x. Denote by π some probability mass value in the segment $[0, 1]$. The per-value accuracy accounts for the probability that the model

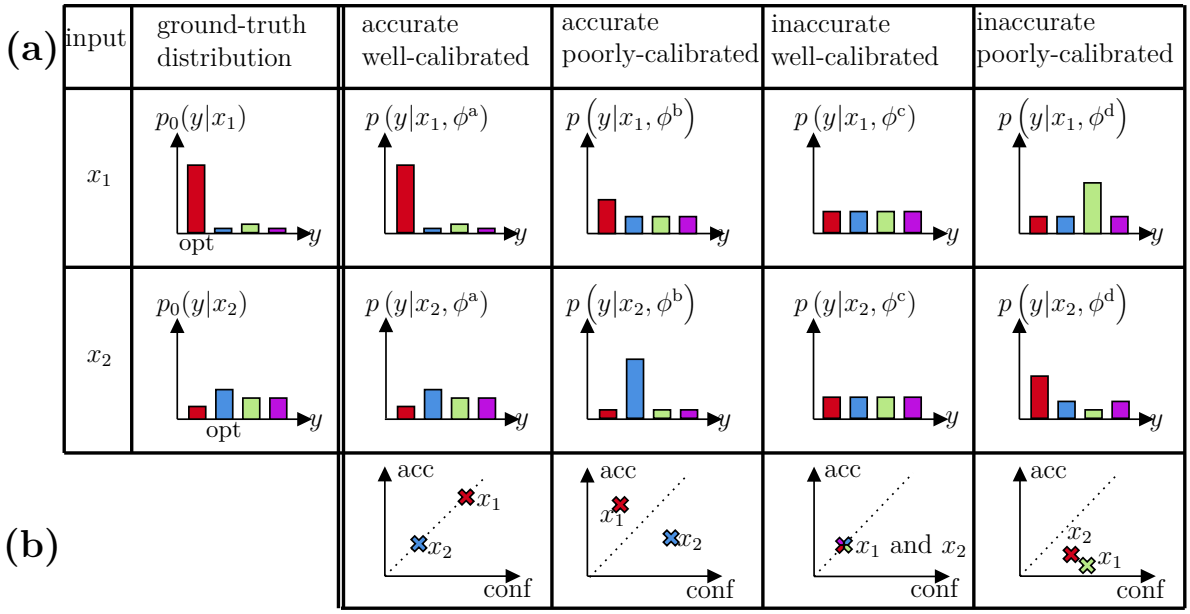


Fig. 2.1 **(a)** Examples of probabilistic predictors for two inputs x_1 and x_2: As compared to the ground-truth distribution in the second column, the first predictor (third column) is accurate, assigning the largest probability to the optimal decision (indicated as "opt" in the second column) and also well calibrated, reproducing the true accuracy of the decision; the second predictor (fourth column) is still accurate, but it is underconfident on the correct decision (for input x_1) and overconfident on the correct decision (for input x_2); the third predictor (fifth column) is not accurate, producing a uniform distribution across all output values, but is well calibrated if the data set is balanced [119]; and the last predictor (sixth column) is both inaccurate and poorly calibrated, providing overconfident decisions. **(b)** Confidence versus accuracy for the decisions made by the corresponding predictors.

predicts the correct label, provided the model is π-confident for the input x, i.e.

$$\mathrm{acc}(\pi|\mathcal{D}) = \mathbb{P}\Big(\mathbf{y} = \hat{y}(\mathbf{x}|\mathcal{D}) \Big| \mathrm{conf}(\mathbf{x}|\mathcal{D}) = \pi\Big). \tag{2.6}$$

As illustrated in the example in Fig. 2.1, accuracy and calibration are distinct criteria, with neither criterion implying the other. It is, for instance, possible to have an accurate predictor that consistently underestimates the accuracy of its decisions, and/or that is overconfident where making incorrect decisions (see fourth column in Fig. 2.1). Conversely, one can have inaccurate predictions that estimate correctly their uncertainty (see fifth column in Fig. 2.1).

Notable calibration metrics include reliability diagram and expected calibration error (ECE) [10], and are reviewed next.

Reliability diagrams plot accuracy with respect to confidence level π, providing a visual depiction of calibration performance. Using this metric, a predictor $p(y|x, \mathcal{D})$ would be considered as perfectly calibrated if it satisfies

$$\text{acc}(\pi|\mathcal{D}) = \pi, \quad \forall \pi \in [0, 1]. \tag{2.7}$$

In contrast, the predictor is considered to be over-confident for confidence level π when the inequality $\text{acc}(\pi|\mathcal{D}) < \pi$ holds, since it exhibits poorer accuracy level than self-reported confidence level, and to be under-confident when the inequality $\text{acc}(\pi|\mathcal{D}) > \pi$ holds.

While the reliability diagram provides a fine-grained evaluation of the calibration performance of a probabilistic predictor, the *expected calibration error* (ECE) provides a single scalar measure obtained as the weighted average of the differences between accuracy and confidence levels across all possible confidence level, i.e.

$$\text{ECE}(\mathcal{D}) = \int_{\pi=0}^{1} \mathbb{P}\Big(\text{conf}(\mathbf{x}|\mathcal{D}) = \pi\Big)\Big|\text{acc}(\pi|\mathcal{D}) - \pi\Big| \, \text{d}\pi. \tag{2.8}$$

A smaller ECE indicates a better calibrated probabilistic predictor.

Since the population distribution is unknown for real life problems, a data-driven estimation of the accuracy (2.5) using a held-out test set $\mathcal{D}^{\text{te}} = \{(x[i], y[i])\}_{i=1}^{N^{\text{te}}}$ with N^{te} samples provides the unbiased estimation

$$\text{acc}(\mathcal{D}^{\text{te}}|\mathcal{D}) = \tfrac{1}{N^{\text{te}}} \sum_{i=1}^{N^{\text{te}}} \mathbb{1}\Big(\hat{y}(x^{\text{te}}[i]|\mathcal{D}) = y^{\text{te}}[i]\Big). \tag{2.9}$$

To visualize *Reliability diagrams* using data sets rather than the population loss, the probability interval $[0, 1]$ is partitioned into M equal length intervals, with the m-th interval $(\frac{m-1}{M}, \frac{m}{M}]$ referred to as the m-th *bin* henceforth. Now let us denote as $\mathcal{B}_m$ the subset of sample indices whose associated confidence level $\text{conf}(x[i]|\mathcal{D})$ lie within the m-th bin, i.e.,

$$\mathcal{B}_m = \Big\{ i \Big| \text{conf}(x^{\text{te}}[i]|\mathcal{D}) \in \Big(\tfrac{m-1}{M}, \tfrac{m}{M}\Big], \text{ with } i = 1, 2, \ldots, N^{\text{te}} \Big\}. \tag{2.10}$$

Evaluating measures within particular bin $\mathcal{B}_m$ amounts for approximation of conditioning on confidence level $\pi \in (\frac{m-1}{M}, \frac{m}{M}]$. Note that the bins partition the data set $\mathcal{D}^{\text{te}}$ into disjoint subsets, since we have $\bigcup_{m=1}^{M} \mathcal{B}_m = \{ 1, 2, \ldots, N^{\text{te}}\}$ and $\mathcal{B}_m \cap \mathcal{B}_{m'} = \emptyset$ for any $m' \neq m$.

Accordingly, the *within-bin accuracy* of the predictor for the m-th bin measures the fraction of within-bin test examples that are correctly detected, i.e.,

$$\mathrm{acc}(\mathcal{B}_m|\mathcal{D}) = \frac{1}{|\mathcal{B}_m|} \sum_{i \in \mathcal{B}_m} \mathbb{1}(y^{\mathrm{te}}[i] = \hat{y}(x^{\mathrm{te}}[i]|\mathcal{D})), \tag{2.11}$$

with $|\mathcal{B}_m|$ denoting the number of total samples in $\mathcal{B}_m$. Furthermore, the *within-bin confidence* of the predictor for the m-th bin is defined as the average within-bin confidence level, i.e.,

$$\mathrm{conf}(\mathcal{B}_m|\mathcal{D}) = \frac{1}{|\mathcal{B}_m|} \sum_{i \in \mathcal{B}_m} p\Big(\hat{y}(x^{\mathrm{te}}[i]|\mathcal{D})\Big|x^{\mathrm{te}}[i], \mathcal{D}\Big). \tag{2.12}$$

A perfectly calibrated predictor $p(y|x, \mathcal{D})$ would have $\mathrm{acc}(\mathcal{B}_m|\mathcal{D}) = \mathrm{conf}(\mathcal{B}_m|\mathcal{D})$ for all $m \in \{1, \ldots, M\}$ in the limit of a sufficiently large test data set, i.e., $N_*^{\mathrm{te}} \to \infty$.

Reliability diagrams plot the accuracy $\mathrm{acc}(\mathcal{B}_m|\mathcal{D})$ and the confidence $\mathrm{conf}(\mathcal{B}_m|\mathcal{D})$ over the binned probability interval $[0, 1]$. Using these metrics, a predictor $p(y|x, \mathcal{D})$ would be considered as perfectly calibrated if it satisfies the equalities $\mathrm{acc}(\mathcal{B}_m|\mathcal{D}) = \mathrm{conf}(\mathcal{B}_m|\mathcal{D})$ for all $m \in \{1, \ldots, M\}$. In contrast, the predictor is considered to be over-confident for bin m when the inequality $\mathrm{conf}(\mathcal{B}_m|\mathcal{D}) > \mathrm{acc}(\mathcal{B}_m|\mathcal{D})$ holds and to be under-confident in bin m when the inequality $\mathrm{conf}(\mathcal{B}_m|\mathcal{D}) < \mathrm{acc}(\mathcal{B}_m|\mathcal{D})$ holds. Fig. 2.2 illustrates reliability diagrams of these kind of predictors.

Finally, the *data-driven ECE* (2.8) is accordingly estimated by

$$\mathrm{ECE}(\mathcal{D}^{\mathrm{te}}|\mathcal{D}) = \frac{1}{N^{\mathrm{te}}} \sum_{m=1}^{M} \Big|\mathcal{B}_m\Big|\Big|\mathrm{acc}(\mathcal{B}_m|\mathcal{D}) - \mathrm{conf}(\mathcal{B}_m|\mathcal{D})\Big|. \tag{2.13}$$

2.3 Frequentist Learning

Given access to training data set $\mathcal{D} = \Big\{z[i] = (x[i], y[i])\Big\}_{i=1}^{N}$ with N examples, frequentist learning finds a model parameter vector $\phi_{\mathcal{D}}^*$ by tackling the following empirical risk minimization (ERM) problem

$$\min_{\phi} \Big\{ L_{\mathcal{D}}(\phi) = -\tfrac{1}{N} \sum_{(x,y) \in \mathcal{D}} \log p(y|x, \phi) = \mathrm{E}_{(\mathbf{x},\mathbf{y}) \sim p_{\mathcal{D}}(x,y)}\big[-\log p(\mathbf{y}|\mathbf{x}, \phi)\big] \Big\}, \tag{2.14}$$

with empirical distribution $p_{\mathcal{D}}(x, y)$ defined by the data set $\mathcal{D}$.

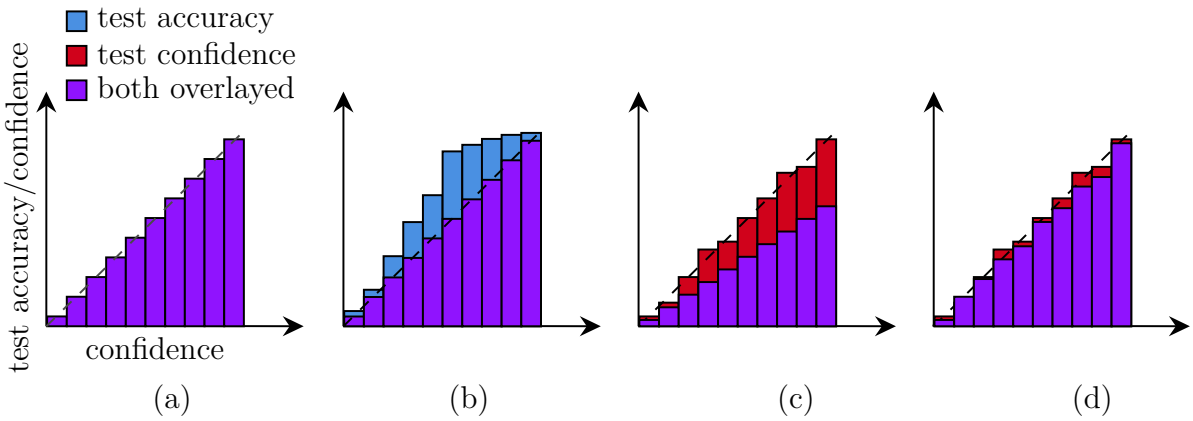


Fig. 2.2 Illustration of reliability diagrams. Provided a data set, the prediction levels are grouped into M bins (10 for this case). The within-bin test confidence and test accuracy are then plotted overlaid on on the other. **(a)** Ideal calibrated for which all predictions coincide over the identity line. **(b)** Underconfident predictor, whose self reporting confidence is lower than the actual accuracy, hence poor-calibrated. **(c)** Overconfident predictor, also poor-calibrated. **(d)** Well-calibrated predictor, having a small gap between test accuracy and confident.

This approach is statistically well-justified [28] if the training loss is a good approximation of the context-specific population loss $L_{\mathrm{p}}(\phi|c^*)$, i.e.,

$$L_{\mathcal{D}}(\phi) \approx L_{\mathrm{p}}(\phi|c^*) = \mathrm{E}_{(\mathbf{x},\mathbf{y})\sim p_0(x,y|c^*)}\Big[-\log p(\mathbf{y}|\mathbf{x},\phi)\Big], \tag{2.15}$$

with ground-truth sampling distribution $p_0(x,y|c^*)$ defined using the (unknown) ground-truth context variable c^* that was used to generate the data set $\mathcal{D}$. In other words, frequentist learning aims at minimizing the population loss assuming a single context variable c^*.

An ideal learner will train a model parameter vector ϕ^* that yields a *well-calibrated* predictor with predictive capabilities matching those of the context-based population law, i.e.,

$$p(y|x,\phi^*) \approx p_0(y|x,c^*) = \frac{p_0(x,y|c^*)}{\sum_{y'\in\mathcal{Y}} p_0(x,y'|c^*)} \tag{2.16}$$

for every pair (x,y). Note that this is a sufficient condition for satisfying the well-calibration condition (2.4). To achieve this via frquentist learning, two assumptions are made. The first assumption, known as *well specificity* [11], assumes the discriminative model is capable enough to express the true unknown context-based likelihood. Mathematically, this amounts

to the assumption of existence of the equality condition

$$p(y|x,\phi^*) = p_0(y|x,c^*) \tag{2.17}$$

for some ϕ^* in the model parameter space Φ. The second assumption regards to the learning process using the data set $\mathcal{D}$, assuming the empirical risk in (2.14) has been obtained from abundant examples so that (2.15) holds and hence the estimation error for the model parameter vector ϕ is low, i.e.,

$$\phi_{\mathcal{D}} \approx \phi^*. \tag{2.18}$$

However, the second assumption is easily broken in the presence of limited training samples [28], which brings inevitable epistemic uncertainty [120, 28] that corresponds to existence of multiple context variables c that are (almost) equally well accounted for generating the finite data set $\mathcal{D}$.

2.4 Bayesian Learning

Bayesian learning addresses epistemic uncertainty by treating the model parameter vector as a random vector $\boldsymbol{\phi}$ with prior distribution $\boldsymbol{\phi} \sim p(\phi)$. Ideally, Bayesian learning updates the prior $p(\phi)$ to produce the posterior distribution $p(\phi|\mathcal{D})$ as

$$p(\phi|\mathcal{D}) \propto p(\phi) \prod_{i=1}^{N} p\Big(y[i]\Big|x[i],\phi\Big) \tag{2.19}$$

and obtains the ensemble predictor for the test point (x,y) by averaging over multiple models, i.e.,

$$p(y|x,\mathcal{D}) = \mathrm{E}_{\boldsymbol{\phi}\sim p(\phi|\mathcal{D})}[p(y|x,\boldsymbol{\phi})]. \tag{2.20}$$

We further assume the ground-truth distribution of the $N+1$ samples of $\{\mathcal{D},(x,y)\}$ to be independent when conditioned on a particular state, i.e.,

$$p_0(\mathcal{D},(x,y)|c) = p_0(x,y|c) \prod_{i=1}^{N} p_0(x[i],y[i]|c).$$

This is justified in the case where there is a state c, stationary over a coherence interval and changes between intervals, which sets the behavior of samples within its coherence interval.

The joint population distribution of the $N+1$ of $\{\mathcal{D}, (x, y)\}$ points follows

$$p_0(\mathcal{D}, (x, y)) = \mathrm{E}_{\mathbf{c}\sim p_0(c)}[p_0(\mathcal{D}, (x, y)|\mathbf{c})]. \tag{2.21}$$

By marginalization, we have the prediction of label y using input x and data set $\mathcal{D}$ by

$$p_0(y|x, \mathcal{D}) = \frac{p_0(\mathcal{D}, (x, y))}{\sum_{y'\in\mathcal{Y}} p_0(\mathcal{D}, (x, y'))}. \tag{2.22}$$

An ideal Bayesian predictor will be *well-calibrated*, meaning its prediction approximates the ground-truth discriminative distribution

$$p(y|x, \mathcal{D}) \approx p_0(y|x, \mathcal{D}) \tag{2.23}$$

for every pair (x, y). Note again that (2.23) is a sufficient condition to satisfy the condition (2.4). In (2.23), unlike (2.16), Bayesian learning admits the possibility of the data set $\mathcal{D}$ being generated from different context variables $\mathbf{c}$ to capture epistemic uncertainty raised from the finite number of training samples N.

As long as the model likelihood is well specified (2.17) and also the prior is well specified, which essentially yields the following equality

$$p(\mathcal{D}, (x, y)) = p_0(\mathcal{D}, (x, y)), \tag{2.24}$$

where the model-based joint distribution is defined as

$$p(\mathcal{D}, (x, y)) = \mathrm{E}_{\boldsymbol{\phi}\sim p(\phi)}\Big[p_0(x)p(y|x, \boldsymbol{\phi}) \prod_{i=1}^{N} \Big(p_0(x[i])p\Big(y[i]\Big|x[i], \boldsymbol{\phi}\Big)\Big)\Big], \tag{2.25}$$

the Bayesian predictor is well-calibrated [11].

The comparison between frequentist learning and Bayesian learning in terms of reliable prediction is summarized in Fig. 2.3.

2.4.1 Approximate Posterior

In practice, as the true posterior distribution is generally intractable due to the normalizing factor in (2.19), approximate Bayesian approaches are considered via VI or MC techniques (see, e.g., [28]).

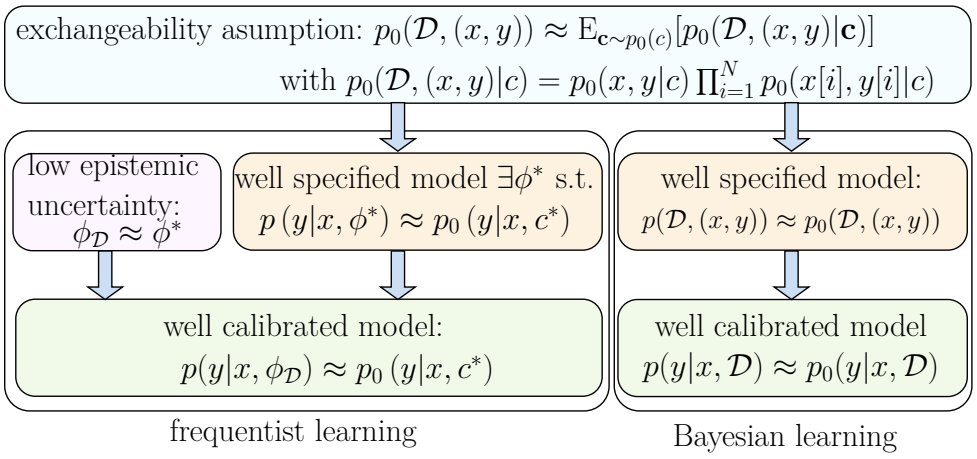


Fig. 2.3 Assumptions required for well-calibrated probabilistic models trained via frequentist and Bayesian learning given training data $\mathcal{D}$. The pair (x,y) represents a test data point. The variable c can be interpreted as a context variable determining the learning task, e.g., a channel realization for communication systems. Frequentist learning requires the likelihood models to be well specified, as well as abundant training data to identify the optimal parameter vector ϕ^*; whereas Bayesian learning only requires the model, including the prior distribution, to be well specified.

Denoting the approximate Bayesian posterior of $p(\phi|\mathcal{D})$ distribution as $q(\phi|\mathcal{D})$, the Bayesian ensemble predictor (2.20) can be approximated as

$$q(y|x,\mathcal{D}) = \mathrm{E}_{\boldsymbol{\phi}\sim q(\phi|\mathcal{D})}[p(y|x,\boldsymbol{\phi})]. \tag{2.26}$$

One common way to tract (2.26) is to use Monte Carlo sampling, namely using R model parameters, each drawn from $\phi[r] \sim q(\phi|\mathcal{D})$ for all $r = 1,\ldots,R$ to obtain

$$q(y|x,\mathcal{D}) \approx \tfrac{1}{R}\sum_{r=1}^{R} p(y|x,\phi[r]), \tag{2.27}$$

as illustrated in Fig. 2.4. The epistemic uncertainty is embedded via disagreement [61] among R models. In Appendix B.3, we detail one of the most popular MC technique, stochastic gradient Langevin dynamics (SGLD) [39].

2.4.2 Why Well-Specificity leads to Well-Calibrated Predictor

Now we detail how assuming well specificity (2.24) leads to well-calibrated Bayesian predictors (2.20). As a special case of (2.24) with one sample less, $p(\mathcal{D}) \approx p_0(\mathcal{D})$ follows immediately.

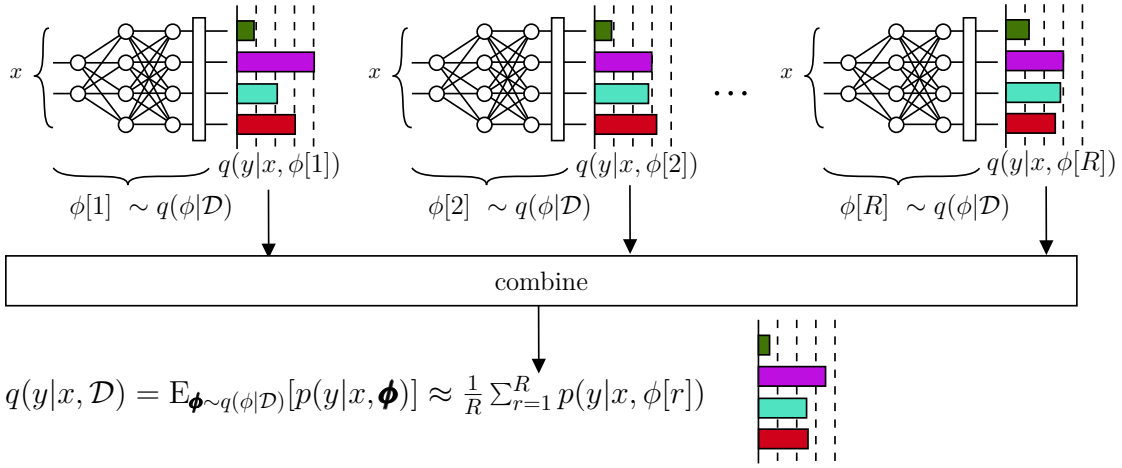


Fig. 2.4 Bayesian neural network yields an approximated ensembling prediction $q(y|x,\mathcal{D})$ via the combination of the R outcomes from the model distribution $q(\phi|\mathcal{D})$.

$$
\begin{aligned}
p\Big(y\Big|x,\mathcal{D}\Big) &= \mathrm{E}_{\boldsymbol{\phi}\sim p(\phi|\mathcal{D})}[p(y|x,\boldsymbol{\phi})] \\
&= \mathrm{E}_{\boldsymbol{\phi}\sim p(\phi)}\left[\frac{p(\mathcal{D}|\boldsymbol{\phi})}{p(\mathcal{D})}p(y|x,\boldsymbol{\phi})\right] \\
&= \frac{p(\mathcal{D},(x,y))}{p_0(x)p(\mathcal{D})} \\
&\approx \frac{p_0(\mathcal{D},(x,y))}{p_0(x)p_0(\mathcal{D})} \\
&= \mathrm{E}_{\mathbf{c}\sim p_0(c)}\left[\frac{p_0(\mathcal{D}|\mathbf{c})}{p_0(\mathcal{D})}p_0(y|x,\mathbf{c})\right] \\
&= \mathrm{E}_{\mathbf{c}\sim p_0(c|\mathcal{D})}[p_0(y|x,\mathbf{c})] \\
&= p_0(y|x,\mathcal{D}).
\end{aligned}
\tag{2.28}
$$

Hence, calibration in Bayesian learning does not hinge on the availability of a large data set, but *only* on the model well specificity.

2.5 Frequentist Meta-Learning

The most prominent shortcoming of conventional learning is its potentially high sample complexity, which translates into the need for a large number of labeled training points, N_τ^{tr}, per task. Meta-learning addresses this issue by transferring knowledge acquired over previous tasks. Specifically, frequentist meta-learning, as proposed in [49], treats an

initialization ξ of the model parameter vector as a hyperparameter vector to be optimized based on the availability of labeled data from t previous tasks.

As a preliminary step, we decompose the available labeled points from each task τ into a disjoint training set $\mathcal{D}_\tau^{\rm tr}$ and test set $\mathcal{D}_\tau^{\rm te}$ as $\mathcal{D}_\tau = \{\mathcal{D}_\tau^{\rm tr}, \mathcal{D}_\tau^{\rm te}\}$. Furthermore, the data sets for all previous t tasks are stacked as $\mathcal{D}_{1:t} = \{\mathcal{D}_\tau\}_{\tau=1}^t$, and similarly for $\mathcal{D}_{1:t}^{\rm te} = \{\mathcal{D}_\tau^{\rm te}\}_{\tau=1}^t$, having a total of $N_{1:t}^{\rm te} = \sum_{\tau=1}^t N_\tau^{\rm te}$ samples. Meta-learning has two phases: meta-training and meta-testing. These are defined next by following the frequentist meta-learning strategy of [49].

Meta-training tackles the bi-level optimization problem

$$\min_{\xi} \quad \tfrac{1}{N_{1:t}^{\rm te}} \sum_{\tau=1}^{t} N_\tau^{\rm te} \mathcal{L}_{\mathcal{D}_\tau^{\rm te}}\Big(\phi_\tau(\mathcal{D}_\tau^{\rm tr}|\xi)\Big) \tag{2.29a}$$

$$\text{s.t.} \quad \phi_\tau(\mathcal{D}_\tau^{\rm tr}|\xi) = \underset{\phi(\xi)}{\rm argmin}\, \mathcal{L}_{\mathcal{D}_\tau^{\rm tr}}(\phi), \quad \tau = 1, \ldots, t. \tag{2.29b}$$

The notation $\phi(\xi)$ in (2.29b) indicates the dependence of the optimizer on the initialization vector ξ, and $\mathcal{L}_{\mathcal{D}}(\phi)$ stands for the empirical loss of model parameter ϕ using data set $\mathcal{D}$, e.g., the log-loss (2.14). By (2.29), the goal of frequentist meta-training is to find a hyperparameter vector ξ such that for any task τ, the optimized model parameter vector $\phi_\tau(\mathcal{D}_\tau^{\rm tr}|\xi)$ fits well the test data set $\mathcal{D}_\tau^{\rm te}$.

Problem (2.29) is addressed via a nested loop optimization involving Gradient Descent (GD)-based inner updates and Stochastic Gradient Descent (SGD)-based outer updates, which are also referred as meta-iterations. The inner loop tackles the inner optimization (2.29b) in a per-task manner, as detailed in the next chapter via (3.6), for a randomly selected subset $\mathcal{T} \subset \{1, \ldots, t\}$ of tasks, which are redrawn independently at each meta-iteration. The outer loop addresses the outer optimization (2.29a) via an SGD step of the meta-loss with learning-rate $\kappa > 0$, i.e.,

$$\xi \leftarrow \xi - \kappa \frac{1}{N_{\mathcal{T}}^{\rm te}} \sum_{\tau \in \mathcal{T}} N_\tau^{\rm te} \nabla_\xi \mathcal{L}_{\mathcal{D}_\tau^{\rm te}}\Big(\phi^{\rm GD}(\mathcal{D}_\tau^{\rm tr}|\xi)\Big), \tag{2.30}$$

based on data from the batch $\mathcal{T}$ of selected tasks, and using the notation $N_{\mathcal{T}}^{\rm te} = \sum_{\tau \in \mathcal{T}} N_\tau^{\rm te}$ for the total samples within the batch of selected tasks. Meta-training updates the initialization vector ξ across multiple meta-iterations. When meeting some stopping criterion, here determined by a predefined number of meta-iterations $I_{\rm meta}$, meta-training stops, and the hyperparameter vector ξ is stored to be used for future learning tasks.

Upon deployment, i.e., during *meta-testing*, the meta-test tasks also include labeled points and unlabeled payload data as the meta-training tasks. Accordingly, each meta-test

device loads the hyperparameter vector ξ for initialization, and produces the adapted model parameter vector $\phi_* = \phi^{\mathrm{GD}}(\mathcal{D}_*^{\mathrm{tr}}|\xi)$, as detailed in the next chapter (3.6), using N_*^{tr} labeled symbols $\mathcal{D}_*^{\mathrm{tr}} = \{(x_*^{\mathrm{tr}}[i], y_*^{\mathrm{tr}}[i])\}_{i=1}^{N_*^{\mathrm{tr}}}$. Then, it applies the learned model to the payload data symbols $\{x_*^{\mathrm{te}}[i]\}_{i=1}^{N_*^{\mathrm{te}}}$ to carry out demodulation or equalization

$$p(y_*^{\mathrm{te}}[i]|x_*^{\mathrm{te}}[i], \phi_*). \tag{2.31}$$

2.6 Set Predictors

A *set predictor* is defined as a set-valued function $\Gamma(\cdot|\mathcal{D}) : \mathcal{X} \to 2^{\mathcal{Y}}$ that maps an input x to a subset of the output domain $\mathcal{Y}$ based on data set $\mathcal{D}$. We denote the size of the set predictor for input x as $|\Gamma(x|\mathcal{D})|$. As illustrated in the example of Fig. 4.1, the set size $|\Gamma(x|\mathcal{D})|$ generally depends on input x, and it can be taken as a measure of the uncertainty of the set predictor.

The performance of a set predictor is evaluated in terms of calibration, or coverage, as well as of inefficiency. *Coverage* refers to the probability that the true label is included in the predicted set; while *inefficiency* refers to the average size $|\Gamma(x|\mathcal{D})|$ of the predicted set. There is clearly a trade-off between two metrics. A conservative set predictor that always produces the entire output space, i.e., $\Gamma(x|\mathcal{D}) = \mathcal{Y}$, would trivially yield a coverage probability equal to 1, but at the cost of exhibiting the worst possible inefficiency of $|\mathcal{Y}|$. Conversely, a set predictor that always produces an empty set, i.e., $\Gamma(x|\mathcal{D}) = \emptyset$, would achieve the best possible inefficiency, equal to zero, while also presenting the worst possible coverage probability equal to zero.

Let us denote a set predictor $\Gamma(\cdot|\cdot)$ for short as Γ. Formally, the *coverage* level of set predictor Γ is the probability that the true output y is included in the prediction set $\Gamma(x|\mathcal{D})$ for a test pair $z = (x, y)$. This can be expressed as $\mathrm{coverage}(\Gamma) = \mathbb{P}\left(\mathbf{y} \in \Gamma(\mathbf{x}|\boldsymbol{\mathcal{D}})\right)$, where the probability $\mathbb{P}(\cdot)$ is taken over the ground-truth joint distribution $p_0(\mathcal{D}, (x, y))$ in (4.2). The set predictor Γ is said to be $(1-\alpha)$-*valid* if it satisfies the inequality

$$\mathrm{coverage}(\Gamma) = \mathbb{P}\Big(\mathbf{y} \in \Gamma(\mathbf{x}|\boldsymbol{\mathcal{D}})\Big) \geq 1 - \alpha. \tag{2.32}$$

When the desired coverage level $1-\alpha$ is fixed by the predetermined target *miscoverage level* $\alpha \in [0, 1]$, we will also refer to set predictors satisfying (2.32) as being well calibrated.

Following the discussion in the previous paragraph, it is straightforward to design a valid, or well-calibrated, set predictor, even for the restrictive case of miscoverage level $\alpha = 0$. This can be, in fact, achieved by producing the full set $\Gamma(x|\mathcal{D}) = \mathcal{Y}$ for all inputs x.

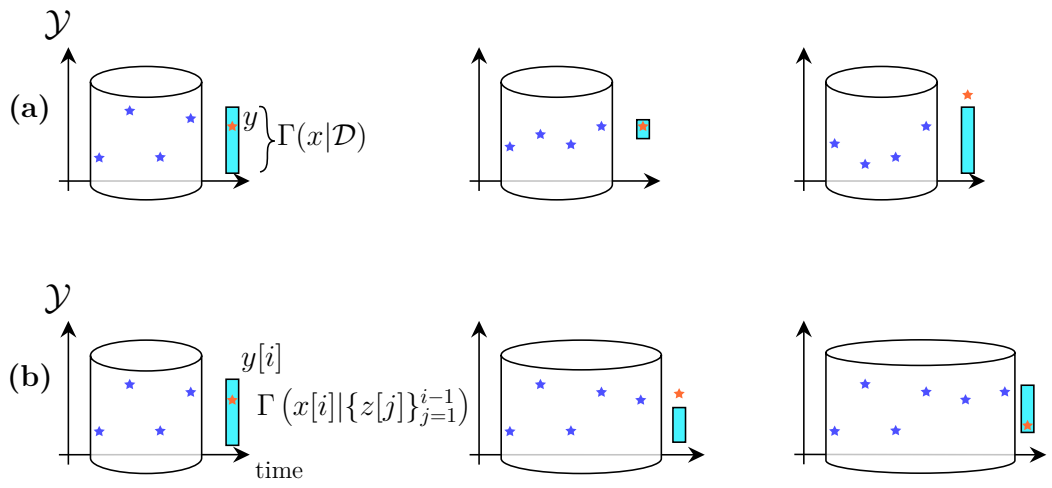


Fig. 2.5 **(a)** The validity condition (2.32) assumed in offline CP is relevant if one is interested in the average performance with respect to realizations $(\boldsymbol{\mathcal{D}}, \mathbf{z}) \sim p_0(\mathcal{D}, z)$ of training set $\mathcal{D}$ and test variable $z = (x, y)$. Input variable x is not explicitly shown in the figure, and the horizontal axis runs over the training examples in $\mathcal{D}$ and the test example z. **(b)** In online CP, the set predictor Γ_i uses its input $x[i]$ and all previously observed pairs $z[1], \ldots, z[i-1]$ with $z[i] = (x[i], y[i])$ to produce a prediction set. The long-term validity (4.16) assumed by online CP is defined as the empirical time-average rate at which the predictor Γ_i includes the true target variable $y[i]$.

One should, therefore, also consider the inefficiency of predictor Γ. The *inefficiency* of set predictor Γ is defined as the average prediction set size

$$\text{inefficiency}(\Gamma) = \mathrm{E}_{(\mathcal{D},x,y)\sim p_0(\mathcal{D},(x,y))}\Big[\big|\Gamma(\mathbf{x}|\boldsymbol{\mathcal{D}})\big|\Big], \tag{2.33}$$

where the average is taken over the data set $\boldsymbol{\mathcal{D}}$ and the test pair $(\mathbf{x}, \mathbf{y})$ following their exchangeable joint distribution $p_0(\mathcal{D}, (x, y))$.

In practice, the coverage condition (2.32) is relevant if the learner produces multiple predictions using independent data set $\mathcal{D}$, and is tested on multiple pairs (x, y). In fact, in this case, the probability in (2.32) can be interpreted as the fraction of predictions for which the set predictor $\Gamma(x|\mathcal{D})$ includes the correct output. This situation, illustrated in Fig. 2.5(a), is quite common in communication systems, particularly at the lower layers of the protocol stack. For instance, the data $\mathcal{D}$ may correspond to pilots received in a frame, and the test point z to a symbol within the payload part of the frame (see Sec. 4.7). While the coverage condition (2.32) is defined under the assumption of a fixed ground-truth distribution $p_0(\mathcal{D}, z)$, in Sec. 4.5 we will allow for temporal distributional shifts and we will focus on validity metrics defined as long-term time averages (see Fig. 2.5(b)).

Chapter 3

Bayesian Active Meta-Learning

In this chapter, we rely on the background provided by Section 3.1, specifically the channel model and soft demodulation and equalization as in Section 3.1.1, with standard frequentist learning and frequentist meta-learning, as in Section 2.3 and Section 2.5 respectively. As an overview of this chapter, we illustrate in Fig. 3.1 a comparison between learning and meta-learning, for both the frequentist and Bayesian learning.

The rest of the chapter is organized as follows. We review in detail Bayesian meta-learning in Section 3.2.2, starting off with Bayesian-learning and then expanding on Bayesian meta-learning. Then, we present Bayesian active meta-learning in Section 3.3. Numerical results are presented in Section 3.4, and Section 3.6 concludes the chapter.

3.1 Channel Model and Background

3.1.1 Channel Model and Soft Demodulation or Equalization

In this chapter, we consider frame-based transmission over a memoryless block fading channel model with constellation $\mathcal{Y}$ and channel output's alphabet $\mathcal{X}$. The channel is characterized by a conditional distribution $p_0(x|y,c)$ of received symbol $x \in \mathcal{X}$ given transmitted symbol $y \in \mathcal{Y}$ and channel state c. In the case of demodulation, we treat the set $\mathcal{Y}$ as discrete; while for equalization we view it as the space of vectors of a certain size. In both cases, we will refer to channel input y as symbol. The channel state c is constant within each frame, and it is independently and identically distributed (i.i.d.) across frames according to an unknown distribution $p(c)$. At frame τ, the transmitter sends a packet consisting of N_τ symbols $y_\tau = \{y_\tau[i]\}_{i=1}^{N_\tau}$. In this chapter a frame is considered as a learning task in the notation of Sec. 2.5. Given the channel state c_τ and the transmitted symbols,

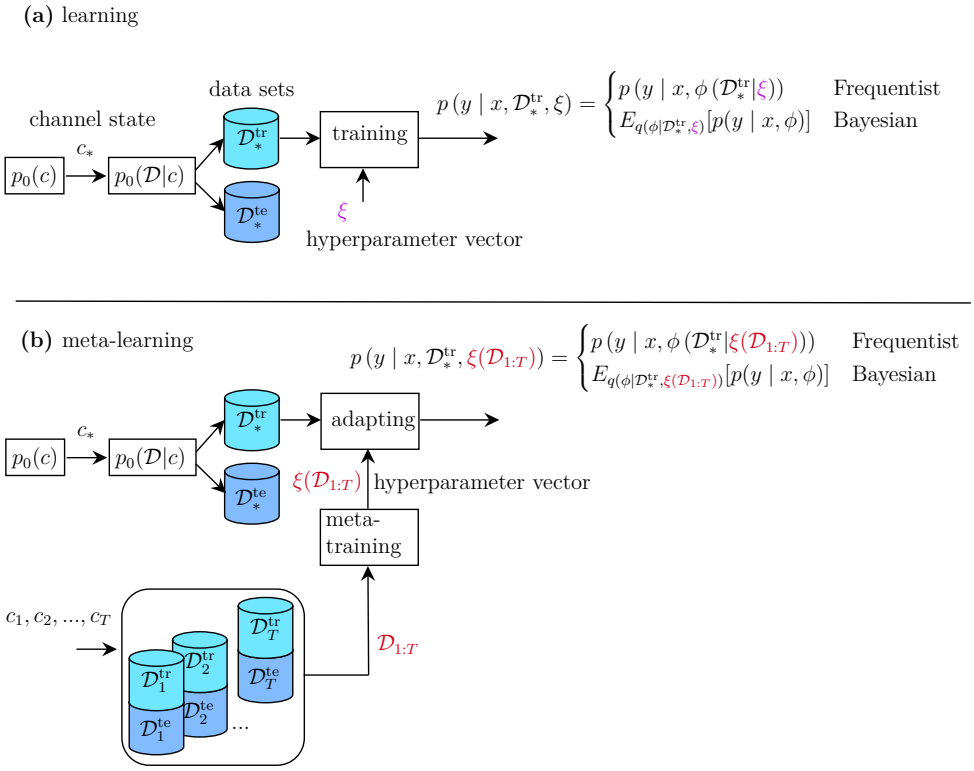


Fig. 3.1 Learning and meta-learning for Frequentist and Bayesian frameworks. **(a)** learning produces a prediction model using a training data set $\mathcal{D}_*^{\text{tr}}$ and a hyperparameter vector ξ. This model is then used over the test inputs of $\mathcal{D}_*^{\text{te}}$. **(b)** meta-learning produces a prediction model using a training data set $\mathcal{D}_*^{\text{tr}}$ and a hyperparameter vector ξ which is trained over available data sets of other T frames $\mathcal{D}_1, \ldots, \mathcal{D}_T$.

collected in a vector x_τ, the received samples $x_\tau = \{x_\tau[i]\}_{i=1}^{N_\tau}$ are conditionally independent and each received i-th sample is distributed as $x_\tau[i] \sim p_0(x_\tau[i]|y_\tau[i], c_\tau)$.

A *soft demodulator/equalizer* is a conditional distribution $p(y|x, \phi)$ that maps channel outputs $x \in \mathcal{X}$ to estimated probabilities for channel input symbol $y \in \mathcal{Y}$. The demodulator/equalizer depends on a vector of parameters ϕ, and is applied separately to each received sample $x[i]$ in a memoryless fashion as $p(y|x_\tau[i], \phi)$. The ideal frame-specific parameter vector ϕ_τ for the frame τ is the one that best approximates the channel conditional distribution $p_0(y_\tau|x_\tau, c_\tau)$, within its model class, obtained from the Bayes rule as

$$p(y_\tau|x_\tau, \phi_\tau) \approx p_0(y_\tau|x_\tau, c_\tau) = \frac{p_0(x_\tau|y_\tau, c_\tau)p_0(y_\tau)}{\sum_{y'_\tau \in \mathcal{Y}} p_0(x_\tau|y'_\tau, c_\tau)p_0(y'_\tau)}, \tag{3.1}$$

where $p_0(y_\tau)$ is the distribution of the input symbol vector y_τ. In practice, as we detail below, the demodulator/equalizer is optimized based on pilot symbols. To simplify the terminology, we will also refer to demodulation/equalization as *prediction* henceforth.

3.1.2 Conventional Data-Driven Demodulators/Equalizers

Pilot-aided schemes utilize available pilot symbols to adapt the predictor $p(y|x, \phi)$ to the unknown channel state c in each frame τ. A typical choice for a predictor is a multi-layer neural-network [121]. With L layers, given received sample x, this class of models produces a vector

$$a(x|\phi) = W_L \cdot f_{W_{L-1},b_{L-1}} \circ \cdots \circ f_{W_1,b_1}(x) + b_L, \tag{3.2}$$

where $\circ$ is the composition operator; the weights $\{W_l\}_{l=1}^L$ and biases $\{b_l\}_{l=1}^L$ define the model parameter vector $\phi := \{W_l, b_l\}_{l=1}^L$ for a total of D parameters; and the function for the l-th layer f_{W_l,b_l} is a linear mapping followed by an entry-wise activation function $h(\cdot)$, i.e., $x_l = f_{W_l,b_l}(x_{l-1}) = h(W_l \cdot x_{l-1} + b_l)$ with $x_0 = x$. In the last, L-th layer, a soft *demodulator* applies the softmax function to vector $a(x|\phi)$, producing the probability distribution

$$\begin{aligned} p(y|x, \phi) &= \Big[\text{softmax}(a(x|\phi))\Big]_y \\ &= \frac{\exp([a(x|\phi)]_y)}{\sum_{y' \in \mathcal{Y}} \exp([a(x|\phi)]_{y'})}, \end{aligned} \tag{3.3}$$

using $[\cdot]_y$ as the y-th element of the vector. In contrast, a soft *equalizer* typically defines the conditional distribution

$$p(y|x, \phi) = \mathcal{N}(y|a(x|\phi), \beta^{-1}), \tag{3.4}$$

where the precision β is fixed. Throughout, we use $\mathcal{N}(y|\mu, \Sigma)$ to indicate the probability density function of a Gaussian vector with mean μ and covariance matrix Σ.

In each frame τ, *conventional learning* optimizes the model parameters ϕ_τ using $N_\tau^{\rm tr}$ i.i.d. pilots $\mathcal{D}_\tau^{\rm tr} = \{(x_\tau^{\rm tr}[i], y_\tau^{\rm tr}[i])\}_{i=1}^{N_\tau^{\rm tr}}$ as training data. Optimization of the prediction aims at minimizing the *training log-loss*

$$\mathcal{L}_{\mathcal{D}_\tau^{\rm tr}}(\phi_\tau) := -\tfrac{1}{N_\tau^{\rm tr}} \sum_{i=1}^{N_\tau^{\rm tr}} \log p(y_\tau^{\rm tr}[i]|x_\tau^{\rm tr}[i], \phi_\tau), \tag{3.5}$$

which amounts to the cross entropy for demodulation (3.3) and the quadratic prediction loss for equalization (3.4). Minimization of (3.5) can be done via *gradient descent* (GD), or stochastic GD (SGD), a variant thereof [122].

GD updates model parameter vector ϕ_τ for I iterations with learning rate $\eta > 0$ starting from an initialization vector ξ. Accordingly, the updated parameters $\phi_\tau := \phi^{\rm GD}(\mathcal{D}_\tau^{\rm tr}|\xi)$ are obtained via the iterations

$$\begin{aligned} \phi_\tau^{(0)} &= \xi, \\ \forall i = 1, \ldots, I: \quad \phi_\tau^{(i)} &\leftarrow \phi_\tau^{(i-1)} - \eta \nabla_{\phi_\tau^{(i-1)}} \mathcal{L}_{\mathcal{D}_\tau^{\rm tr}}(\phi_\tau^{(i-1)}), \\ \phi^{\rm GD}(\mathcal{D}_\tau^{\rm tr}|\xi) &= \phi_\tau^{(I)}. \end{aligned} \tag{3.6}$$

The resulting prediction for a test input-output pair $(x_\tau^{\rm te}[i], y_\tau^{\rm te}[i])$ is given as

$$p(y_\tau^{\rm te}[i]|x_\tau^{\rm te}[i], \phi^{\rm GD}(\mathcal{D}_\tau^{\rm tr}|\xi)). \tag{3.7}$$

3.2 From Bayesian Learning to Bayesian Meta-Learning

3.2.1 Bayesian Learning

Bayesian learning treats the model parameter vector ϕ_τ for some frame τ as a random vector, rather than as a deterministic optimization variable as in frequentist learning framework. As illustrated in Fig. 3.2, instead of producing a single demodulator parameters $\phi_\tau = \phi^{\rm GD}(\mathcal{D}_\tau^{\rm tr}|\xi)$ as in (3.6), Bayesian learning produces a distribution $p(\phi_\tau|\mathcal{D}_\tau^{\rm tr}, \xi)$ over the space of the demodulator parameters ϕ_τ. This distribution is computed based on training data $\mathcal{D}_\tau^{\rm tr}$ and on *predetermined* prior distribution $p(\phi_\tau|\xi)$, which depends in turn on the hyperparameter vector ξ, also fixed *a priori*.

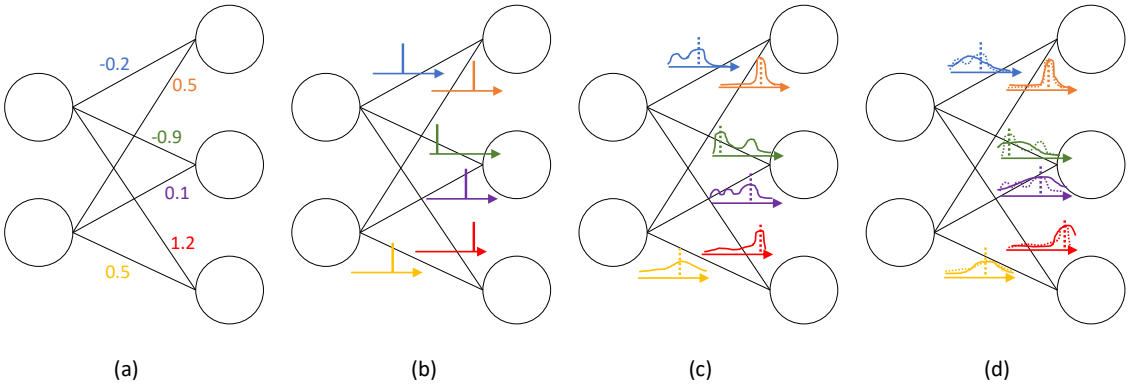


Fig. 3.2 Network weights in frequentist and Bayesian learning: (a) in frequentist learning, each weight is described by a scalar value; (b) the scalar value can be viewed as random variable having a degenerated probabilistic distribution concentrated at a simple prior; (c) in Bayesian learning, the weights are assigned a probability distribution, which, unlike the frequentist point estimate (dashed vertical line), provides information about the uncertainty on the weight; (d) in variational inference (VI), the posterior is approximated with a parameter distribution.

Having obtained the distribution $p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi)$, the *ensemble prediction* of a test point $(x_\tau^{\rm te}[i], y_\tau^{\rm te}[i])$ is given by the ensemble average of the predictions $p(y_\tau^{\rm te}[i]|x_\tau^{\rm te}[i],\phi_\tau)$ with random vector ϕ_τ having distribution $p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi)$, i.e.,

$$p\Big(y_\tau^{\rm te}[i]\Big|x_\tau^{\rm te}[i],\mathcal{D}_\tau^{\rm tr},\xi\Big)=\mathrm{E}_{\boldsymbol{\phi}_\tau\sim p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi)}\Big[p\Big(y_\tau^{\rm te}[i]\Big|x_\tau^{\rm te}[i],\boldsymbol{\phi}_\tau\Big)\Big]. \tag{3.8}$$

The frequentist prediction (2.31), reviewed in the previous section, can be viewed as a special case in which one is limited to the choice $p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi)=\delta(\phi_\tau-\phi^{\rm GD}(\mathcal{D}_\tau^{\rm tr}|\xi))$, with $\delta(\cdot)$ indicating the Dirac Delta. With this choice, the distribution $p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi)$ is concentrated at one point, namely the GD solution (3.6). The frequentist approach is therefore inherently limited in its capacity to express uncertainty on the model parameters due to limited data.

Ideally, the distribution $p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi)$ should be obtained as the posterior distribution

$$p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi)\propto p(\phi_\tau|\xi)p(\mathcal{D}_\tau^{\rm tr}|\phi_\tau), \tag{3.9}$$

where $p(\mathcal{D}_\tau^{\rm tr}|\phi_\tau)=\prod_{i=1}^{N_\tau^{\rm tr}}p(y_\tau^{\rm tr}[i]|x_\tau^{\rm tr}[i],\phi_\tau)$ is the likelihood function for the training data. However, computing the posterior $p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi)$ in (3.9) is generally intractable for high dimensional vector ϕ_τ.

To address this challenge, we follow VI and introduce a *variational distribution* approximation

$$q(\phi_\tau|\varphi_\tau) \approx p(\phi_\tau|\mathcal{D}_\tau^{\text{tr}}, \xi), \tag{3.10}$$

which depends on a variational parameter vector φ_τ. A typical choice is given by the Gaussian mean-field approximation [27] which can be expressed as

$$q(\phi_\tau|\varphi_\tau) = \mathcal{N}(\phi_\tau|\nu_\tau, \text{Diag}(\exp(2\varrho_\tau))), \tag{3.11}$$

with variational parameter vector $\varphi_\tau = [\nu_\tau^\top, \varrho_\tau^\top]^\top$, and the exponent function is applied element-wise. The variational parameter vector includes the mean vector $\nu_\tau \in \mathbb{R}^D$ and the vector of the logarithm of the standard deviations $\varrho_\tau \in \mathbb{R}^D$ for the Gaussian random vector $\boldsymbol{\phi}_\tau$. Note that vector ϱ_τ models uncertainty in the model parameter space. The Gaussian expression (3.11) is termed "mean-field" since the distribution is approximated via factorization across the elements of the random vector, hence the mean. This practice is common in Bayesian networks [29, 123, 30, 59], as well as in other fields of study such as statistical physics and probability theory, due to the dimension reduction and enhanced tractability. While this simplistic variational approximation may impose limitations such as an irreducible bias, it is expressive enough to reflect uncertainty via the variance of each variable. More complex approximations can be considered, as long as they are differentiable with the variational parameter vector. Details on mean-field variational inference can be found in [28, Chapter 10.6].

To describe VI, we will use the Kullback-Liebler (KL) divergence $\text{KL}(q(z)||p(z))$ [124], which is a measure of the distance between two distributions $q(z)$ and $p(z)$. It is defined as the average of the log-likelihood ratio $\log(q(z)/p(z))$ as

$$\text{KL}(q(z)||p(z)) = \text{E}_{\mathbf{z}\sim q(z)}\Big[\log\Big(\frac{q(\mathbf{z})}{p(\mathbf{z})}\Big)\Big]. \tag{3.12}$$

VI-based Bayesian learning prescribes that the variational parameter vectors φ_τ be obtained via the minimization of the KL divergence $\text{KL}(q(\phi_\tau|\varphi)||p(\phi_\tau|\mathcal{D}_\tau^{\text{tr}}, \xi))$ between the variational distribution $q(\phi_\tau|\varphi)$ and the posterior distribution $p(\phi_\tau|\mathcal{D}_\tau^{\text{tr}}, \xi)$. This problem can be equivalently formulated as the minimization [27, 122]

$$\varphi_\tau = \underset{\varphi}{\text{argmin}}\, F_{\mathcal{D}_\tau^{\text{tr}}}(\varphi|\xi), \tag{3.13}$$

where the *variational free energy* [125] is defined as

$$\begin{aligned} F_{\mathcal{D}_\tau^{\rm tr}}(\varphi_\tau|\xi) &= N_\tau^{\rm tr}\mathrm{E}_{\boldsymbol{\phi}_\tau\sim q(\phi_\tau|\varphi_\tau)}[\mathcal{L}_{\mathcal{D}_\tau^{\rm tr}}(\boldsymbol{\phi}_\tau)] + \mathrm{KL}(q(\phi_\tau|\varphi_\tau)||p(\phi_\tau|\xi)) \\ &= N_\tau^{\rm tr} L_{\mathcal{D}_\tau^{\rm tr}}(\varphi_\tau) + \mathrm{KL}(q(\phi_\tau|\varphi_\tau)||p(\phi_\tau|\xi)). \end{aligned} \tag{3.14}$$

The variational free energy (3.14) is a sum of two terms, the first, $L_{\mathcal{D}_\tau^{\rm tr}}(\varphi_\tau)$, which we have defined as the expectation of loss function $\mathcal{L}_{\mathcal{D}_\tau^{\rm tr}}(\phi_\tau)$ (3.5) over variational distribution $q(\phi_\tau|\varphi_\tau)$, i.e.,

$$L_{\mathcal{D}_\tau^{\rm tr}}(\varphi_\tau) = \mathrm{E}_{\boldsymbol{\phi}_\tau\sim q(\phi_\tau|\varphi_\tau)}[\mathcal{L}_{\mathcal{D}_\tau^{\rm tr}}(\boldsymbol{\phi}_\tau)]. \tag{3.15}$$

represents an average training loss. In contrast, the second term in (3.14) is an information-theoretic regularizer that restricts the variational distribution to be close to the prior distribution. Note that, if the variational distribution has ability to express the posterior distribution in (3.9), the minimizer of the problem (3.13) becomes the Bayesian posterior $p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi)$, since the KL divergence $\mathrm{KL}(q(\phi_\tau|\varphi)||p(\phi_\tau|\mathcal{D}_\tau^{\rm tr},\xi))$ is minimized (and it equals zero) when the two distributions are the same. Other regularizers can be considered for different settings, for examples the total variation distance, or the mutual information which can be used to bound generalization error of a learning algorithm [126].

A typical choice for the prior distribution $p(\phi_\tau|\xi)$ is the Gaussian distribution. In this case, we have

$$p(\phi_\tau|\xi) = \mathcal{N}(\phi_\tau|\nu, \mathrm{Diag}(\exp(2\varrho))), \tag{3.16}$$

which is defined by the hyperparameter vector $\xi = [\nu^\top, \varrho^\top]^\top$, where $\nu \in \mathbb{R}^D$ and $\varrho \in \mathbb{R}^D$ stand for the mean and logarithm of the standard deviation vector of the Gaussian random vector $\boldsymbol{\phi}_\tau$.

Assuming the Gaussian variational distribution in (3.11) and the Gaussian prior (3.16), the regularizer term in (3.14) can be computed in closed-form as

$$\mathrm{KL}(q(\phi_\tau|\varphi_\tau)||p(\phi_\tau|\xi)) = \tfrac{1}{2}\sum_{d=1}^{D}\left(2(\varrho[d]-\varrho_\tau[d]) + \frac{\exp(2\varrho_\tau[d]) + (\nu_\tau[d]-\nu[d])^2}{\exp(2\varrho[d])} - 1\right),$$

which is a differentiable function for φ_τ.

With these choices of variational posterior and prior, problem (3.13) can be addressed via gradient-descent methods by using the reparametrization trick [127]. This is done by writing the random model parameter vector $\boldsymbol{\phi}_\tau \sim q(\phi_\tau|\varphi_\tau)$ as $\boldsymbol{\phi}_\tau = \nu_\tau + \exp(\varrho_\tau) \odot \mathbf{e}$, with random vector $\mathbf{e} \sim \mathcal{N}(0, I_D)$ and $\odot$ being the element-wise multiplication. An estimate of the gradient of the objective (3.15) using the reparametrization trick is done with the

aid of R drawn independently samples of the standard normal Gaussian random vector $\mathbf{e}$, and differentiating the resulting empirical estimate of (3.15).

Algorithm 1: `Reparametrization Trick` [127]

Inputs : $\mathcal{G}(\cdot)$ = a function over vector ϕ_τ
φ_τ = variational parameter
Parameters : R = ensemble size
Output : $\hat{G}(\varphi_\tau)$ = approximation of $\mathrm{E}_{\boldsymbol{\phi}_\tau \sim q(\phi_\tau|\varphi_\tau)}[\mathcal{G}(\boldsymbol{\phi}_\tau)]$

for $r = 1, \ldots, R$ **do**
  Draw $e_{\tau,r} \sim \mathcal{N}(0, I_D)$
  $\phi_{\tau,r}(\varphi_\tau, e_{\tau,r}) \leftarrow \nu_\tau + \exp(\varrho_\tau) \odot e_{\tau,r}$
return $\hat{G}(\varphi_\tau) \leftarrow \frac{1}{R}\sum_{r=1}^{R} \mathcal{G}(\phi_{\tau,r}(\varphi_\tau, e_{\tau,r}))$

Specifically, we estimate the free energy in (3.14) by replacing the training loss $L_{\mathcal{D}_\tau^{\mathrm{tr}}}(\varphi_\tau)$ with the empirical estimate

$$\hat{L}_{\mathcal{D}_\tau^{\mathrm{tr}}}(\varphi_\tau) = \frac{1}{R}\sum_{r=1}^{R} \mathcal{L}_{\mathcal{D}_\tau^{\mathrm{tr}}}\Big(\nu_\tau + \exp(\varrho_\tau) \odot \mathbf{e}_{\tau,r}\Big), \tag{3.17}$$

obtained by drawing samples $e_{\tau,r} \sim \mathcal{N}(0, I_D)$ for $r = 1, 2, \ldots, R$. This yields the estimated free energy

$$\hat{F}_{\mathcal{D}_\tau^{\mathrm{tr}}}(\varphi_\tau|\xi) = N_\tau^{\mathrm{tr}} \hat{L}_{\mathcal{D}_\tau^{\mathrm{tr}}}(\varphi_\tau) + \mathrm{KL}(q(\phi_\tau|\varphi_\tau)||p(\phi_\tau|\xi)). \tag{3.18}$$

This is a special case of Algorithm 1 with input $\mathcal{G}(\phi_\tau) = \mathcal{L}_{\mathcal{D}_\tau^{\mathrm{tr}}}(\phi_\tau)$. The function (3.18) can be directly differentiated and used in SGD updates.

Once the variational parameter φ_τ is inferred using Bayesian training, ensemble prediction for a payload data symbol $(x_\tau^{\mathrm{te}}[i], y_\tau^{\mathrm{te}}[i])$ can be obtained via (3.8) by replacing $p(\phi_\tau|\mathcal{D}_\tau^{\mathrm{tr}}, \xi)$ with $q(\phi_\tau|\varphi_\tau)$ to yield the ensemble predictor

$$p(y_\tau^{\mathrm{te}}[i]|x_\tau^{\mathrm{te}}[i], \varphi_\tau) = \mathrm{E}_{\boldsymbol{\phi}_\tau \sim q(\phi_\tau|\varphi_\tau)}\Big[p(y_\tau^{\mathrm{te}}[i]|x_\tau^{\mathrm{te}}[i], \boldsymbol{\phi}_\tau)\Big]. \tag{3.19}$$

Practically, it uses Monte Carlo sampling with R model vectors, producing the approximated soft predictor $\hat{p}(y_\tau^{\mathrm{te}}[i]|x_\tau^{\mathrm{te}}[i], \varphi_\tau)$ via Algorithm 1 with $\mathcal{G}(\phi_\tau) = p(y_\tau^{\mathrm{te}}[i]|x_\tau^{\mathrm{te}}[i], \phi_\tau)$.

3.2.2 Bayesian Meta-Learning

While conventional Bayesian learning assumes that the random model parameter vector $\boldsymbol{\phi}_\tau$ has a fixed prior distribution $p(\phi_\tau|\xi)$ parametrized by a predefined hyperparameter vector ξ, Bayesian meta-learning leverages the stronger assumption that there is a shared prior

distribution $p(\phi_\tau|\xi)$ across all frames that can be optimized through a hyperparameter vector ξ.

In this section, we formulate Bayesian meta-learning by following *empirical Bayes* [128], with the aim of selecting a distribution $p(\phi_\tau|\xi)$ that provides a useful prior for the design of the predictor on new frames. Mathematically, Bayesian meta-training optimizes over the hyperparameter vector ξ by addressing the bi-level problem

$$\min_{\xi} \quad \frac{1}{N^{\text{te}}_{1:t}} \sum_{\tau=1}^{t} N^{\text{te}}_{\tau} \mathrm{E}_{\boldsymbol{\phi}_\tau \sim q(\phi_\tau|\varphi_\tau(\mathcal{D}^{\text{tr}}_\tau|\xi))} \Big[\mathcal{L}_{\mathcal{D}^{\text{te}}_\tau}(\boldsymbol{\phi}_\tau) \Big] \tag{3.20a}$$

$$\text{s.t.} \quad \varphi_\tau(\mathcal{D}^{\text{tr}}_\tau|\xi) = \operatorname*{argmin}_{\varphi} F_{\mathcal{D}^{\text{tr}}_\tau}(\varphi|\xi), \quad \tau = 1, \ldots, t. \tag{3.20b}$$

Problem (3.20) chooses the hyperparameter vector ξ that minimizes the average test loss on the meta-training frames $\tau \in \{1, \ldots, t\}$ that is obtained with the variational posterior via (3.13). The subproblems in (3.20b) correspond to Bayesian learning applied separately to each frame as explained in Section 3.2.1. An illustration of all the quantities involved in problem (3.20) can be found in Fig. 3.3 by using the formalism of Bayesian networks [129].

To address problem (3.20) in a tractable manner, we apply the reparametrization trick for both outer (3.20a) and inner optimization (3.20b) by following the same steps described in Section 3.2.1. Details on the optimization can be found in Algorithm 2. In short, the inner loop updates the frame-specific variational parameters φ_τ by minimizing the approximated free energy (3.18) separately for each frame τ within a mini-batch $\mathcal{T}$ via GD (dashed blue line in Fig. 3.3b). Following [59, 30], the prior's parameter vector ξ plays two roles in the inner loop, namely *(i)* as the initialization for the inner GD update in Algorithm 2 line 9; and *(ii)* as the regularizer for the same update via the prior $p(\phi_\tau|\xi)$. The outer optimization (3.20a) is addressed via SGD to minimize the average log-likelihood for test set using Algorithm 1 with $\mathcal{G}(\phi_\tau) = \mathcal{L}_{\mathcal{D}^{\text{te}}_\tau}(\phi_\tau)$, shown as dashed green line in Fig. 3.3b.

After obtaining meta-trained hyperparameter ξ, meta-testing takes place, starting with the adaptation of the variational parameter $\varphi_*(\mathcal{D}^{\text{tr}}_*|\xi)$ via (3.20b) using the available pilot data $\mathcal{D}^{\text{tr}}_*$ at the current frame, to obtain ensemble prediction

$$p(y^{\text{te}}_*[i]|x^{\text{te}}_*[i], \varphi_*) = \mathrm{E}_{\boldsymbol{\phi}_* \sim q(\phi_*|\varphi_*)} \Big[p(y^{\text{te}}_*[i]|x^{\text{te}}_*[i], \boldsymbol{\phi}_*) \Big], \tag{3.21}$$

as done in (3.19). Bayesian meta-learning is illustrated comparatively to meta-learning in Fig. 3.4.

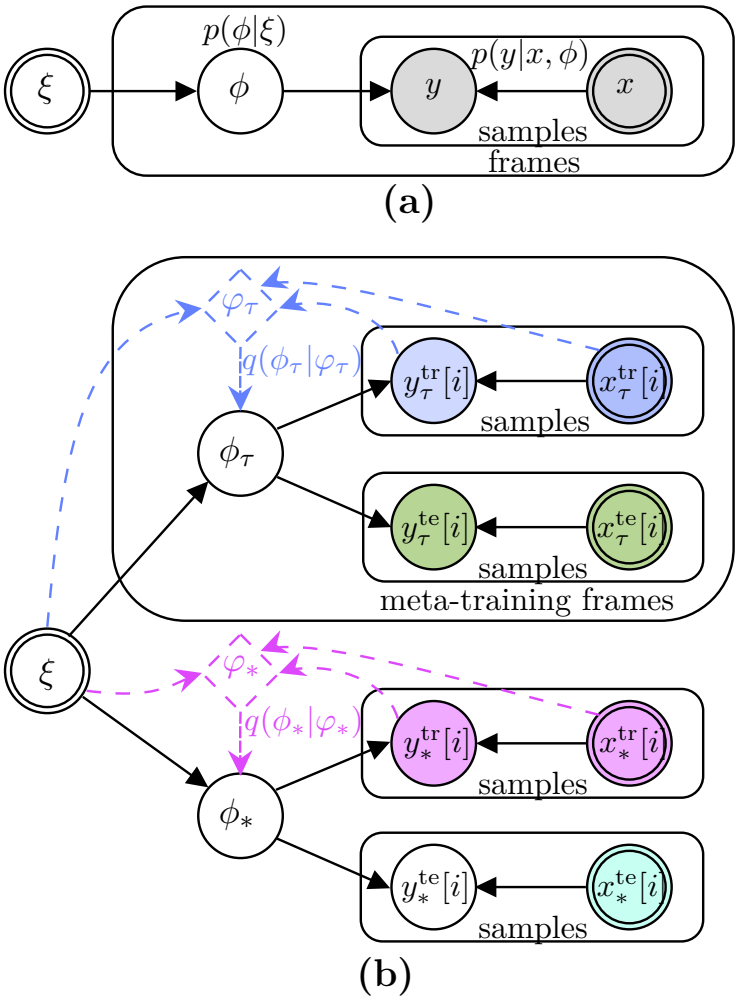


Fig. 3.3 Probabilistic graphical model (Bayesian network) [129] for Bayesian meta-learning. Circles represent random variables; double-lined circles represent deterministic variables or (hyper)parameters; shaded circles represent observations; dashed diamonds represent variational parameter vectors; and plaques indicate multiple instances (the outer plaques represent frames, whereas the inner represent multiple sample, e.g., symbols across time): (a) High level representation, assuming a prior $p(\phi|\xi)$ and predictor $p(y|x,\phi)$; (b) Model using the train/test splits, with variational inference $q(\phi_\tau|\varphi_\tau) \approx p(\phi_\tau|\mathcal{D}_\tau^{\mathrm{tr}},\xi)$ indicated as dashed arrows.

3.2.3 Computational Complexity

We now briefly elaborate on the complexity of meta-learning by analyzing the complexity of meta-training and of meta-testing. To this end, let us define as C the complexity of obtaining the probability $p(y|x,\phi)$ for a data sample (x,y). This baseline complexity depends on the model dimensionality, and it accounts for the amount of time needed to carry

Algorithm 2: `Bayesian Meta-Training`

Inputs : $\mathcal{D}_{1:t}$ = labeled data sets of t meta-training frames
Parameters : B = number of frames per meta-update batch
I = number of inner update steps
η, κ = inner/outer updates learning rates
Output : ξ = learned hyperparameter vector

initialize ξ
while *meta-learning not done* **do**
  $\mathcal{T} \leftarrow$ random batch of B frames
  for $\tau \in \mathcal{T}$ **do**
    randomly divide $\mathcal{D}_\tau = \{\mathcal{D}_\tau^{\text{tr}}, \mathcal{D}_\tau^{\text{te}}\}$
    ◁ *frame-specific update* ▷
    $\varphi_\tau^{(0)} \leftarrow \xi$
    for $i = 1, 2, \ldots, I$ *inner update steps* **do**
      $\varphi_\tau^{(i)} \leftarrow \varphi_\tau^{(i-1)} - \frac{\eta}{N_\tau^{\text{tr}}} \nabla_{\varphi_\tau} \hat{F}_{\mathcal{D}_\tau^{\text{tr}}}(\varphi_\tau^{(i-1)}|\xi)$ using (3.18)
    $\varphi^{\text{GD}}(\mathcal{D}_\tau^{\text{tr}}|\xi) \leftarrow \varphi_\tau^{(I)}$
  ◁ *meta-update* ▷
  $\xi \leftarrow \xi - \kappa \frac{1}{N_{\mathcal{T}}^{\text{te}}} \sum_{t \in \mathcal{T}} N_\tau^{\text{te}} \nabla_\xi \hat{L}_{\mathcal{D}_\tau^{\text{te}}}(\varphi^{\text{GD}}(\mathcal{D}_\tau^{\text{tr}}|\xi))$
return ξ

out the forward pass on the neural network implementing the model $p(y|x, \phi)$. Accordingly, as seen in Table 3.1, the per-data point complexity of meta-testing equals C for frequentist learning, and CR^{te} for Bayesian learning, where R^{te} is the size of the ensemble used for inference.

The meta-update step (2.30) is unrolled via backpropagating I times, as each model parameter $\phi_\tau^{(i)}$ is connected via i local GD updates starting from initialization ξ (3.6). While mathematically the Jacobian should be considered for each step, the overall update rule requires only I Hessian-vector products (HVP). The full unrolling trace is detailed in [49]. The complexity of computing the first-order gradient via backpropagation per-sample is given by $G_1 C$, with G_1 being a constant in the range between 2 and 5 [28, 130]. Each HVP has a complexity of the order $G_2 G_1 C$, where constant G_2 is also between 2 to 5 [49, Appendix A],[131, Appendix C]. Assume that all tasks have data sets of equal size, i.e., $N_\tau^{\text{tr}} = N^{\text{tr}}$ and $N_\tau^{\text{te}} = N^{\text{te}}$ for any task τ. Therefore, for each meta-training iteration, for a batch of B tasks with I local updates, the complexity of the frequentist meta-update (2.30) is of the order

$$B\Big(\underbrace{IN^{\text{tr}} G_1 C}_{\text{frame-specific update}} + \underbrace{IN^{\text{tr}} G_2 G_1 C}_{\text{HVPs in meta-update}} + \underbrace{N^{\text{te}} G_1 C}_{\text{gradient in meta-update}} \Big). \tag{3.22}$$

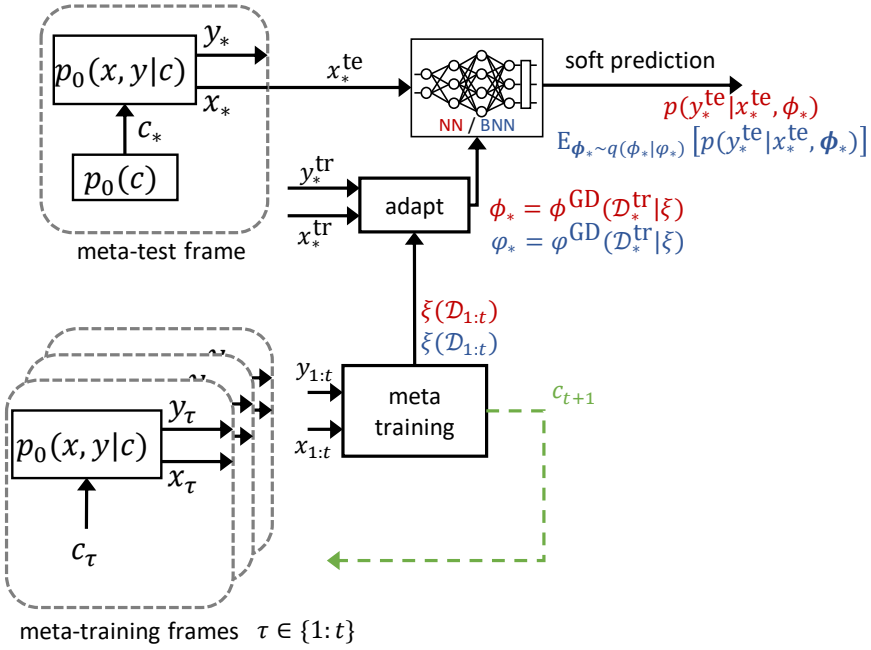


Fig. 3.4 Bayesian meta-learning (blue) as compared to frequentist meta-learning (red). The frequentist predictor uses a single predictor, depicted as a neural network (NN), whereas Bayesian meta-learning uses an ensemble of predictors, e.g., a Bayesian NN (BNN). The dashed line represents the operation of the active meta-learning introduced in Section 3.3. The data for each frame is generated by following the distribution $p_0(x, y|c) = p_0(x)p_0(y|x, c)$, with input distribution $p_0(x)$ and conditional distribution $p_0(y|x, c)$ for channel state c.

As calculating the HVPs may burden the computational complexity, a first-order approximation named first-order model-agnostic meta-learning was suggested in [43], where all second-derivative terms are ignored, leading to a reduced computational complexity without the contributions of the HVPs of the meta-update in (3.22). While there are some reports the performance degradation may be limited while using the approximation, in our simulations we use the full, second-order-based, meta-learning.

For Bayesian meta-learning, the complexity increases linearly with the training ensemble size that is used for estimating the loss functions in (3.20a) and (3.20b). Note that the impact of the size R^{tr} of the training ensemble used for meta-training is different from the size R^{te} used for inference, as the first determines the variance of the stochastic loss functions, while the latter determines the quality of Bayesian prediction (see, e.g., [132] and references therein). Ignoring the constant cost of differentiating the KL term in the

free energy and for sampling from the Gaussian distribution, the complexity analysis is summarized in Table 3.1.

Table 3.1 Computational complexity of frequentist and Bayesian meta-learning. (See text in Sec. 3.2.3 for details)

	inference [per-test sample]	meta-training [per-meta-iteration]
frequentist	C	$BG_1C\Big(IN^{\text{tr}}(G_2+1)+N^{\text{te}}\Big)$
Bayesian	CR^{te}	$BG_1C\Big(IN^{\text{tr}}(G_2+1)R^{\text{tr}}+N^{\text{te}}R^{\text{te}}\Big)$

3.3 Bayesian Active Meta-Learning

In the previous sections, we have considered a passive meta-learning setting in which the meta-learner is given a number of meta-training data sets, each corresponding to a different channel state c. In this section, we study the situation in which the meta-learner has access to a simulator that can be used to generate random data sets for any channel state c via the channel $p_0(x|y,c)$. The problem of interest is to minimize the use of the simulator by actively selecting the channels $\{c_\tau\}$ for which meta-training data is generated. To this end, we devise a sequential approach, whereby the meta-learner optimizes the next channel state c_{t+1}, given all t meta-training data sets of frames $\tau = 1, \ldots, t$.

At the core of the proposed active meta-learning strategy, are mechanisms used by the meta-learner to discover model parameter vectors ϕ that have been underexplored so far, and to relate model parameter vector ϕ to a channel state.

3.3.1 Active Selection of Channel States

After having collected t meta-training data sets $\mathcal{D}_{1:t} = \{\mathcal{D}_\tau\}_{\tau=1}^t$, the proposed active meta-learning scheme selects the next channel state, c_{t+1}, to use for the generation of the $(t+1)$-th meta-training data set $\mathcal{D}_{t+1}$. We adopt the general principle of maximizing the amount of "knowledge" that can be extracted from the data set associated with selected channel c_{t+1}, when added to the t available data sets $\mathcal{D}_{1:t}$. This is done via the following three steps: *(i)* searching in the space of model parameter vectors for a vector ϕ_{t+1} that is most "surprising" given the available meta-training data $\mathcal{D}_{1:t}$; *(ii)* translating the selected model parameter vector ϕ_{t+1} into a channel c_{t+1}; and *(iii)* generating data set $\mathcal{D}_{t+1}$ by using the simulator with input c_{t+1}.

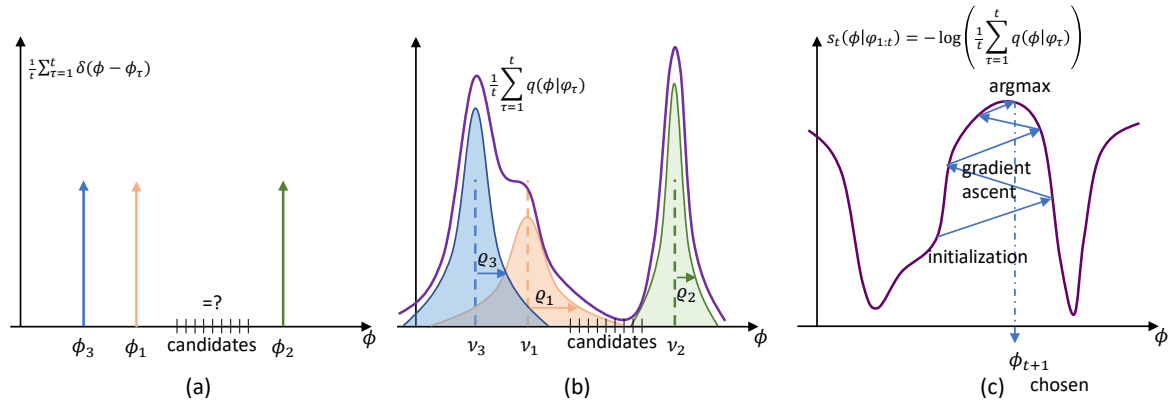


Fig. 3.5 Illustration of how model parameter vectors are scored to enable active meta-learning provided $t = 3$ meta-training sets. (a) Frequentist meta-learning relies on point estimates, and is hence unable to score as-of-yet unexplored model parameters; (b) Bayesian meta-learning can associate a score to each model parameter vector ϕ based on the variational distributions $\left\{q(\phi|\varphi_\tau)\right\}$ evaluated in the previously observed frames $\tau = 1, \ldots, t$; (c) The scoring function can be maximized to obtain the next model parameter vector ϕ_{t+1} as the most "surprising" one.

As illustrated in Fig. 3.5, in step *(i)*, we adopt the scoring function introduced in [65], i.e.,

$$s_t(\phi|\varphi_{1:t}) := -\log\left(\frac{1}{t}\sum_{\tau=1}^{t} q(\phi|\varphi_\tau)\right) \tag{3.23}$$

in order to select the next model parameter vector as

$$\phi_{t+1} = \underset{\phi}{\operatorname{argmax}}\, s_t(\phi|\varphi_{1:t}). \tag{3.24}$$

The criterion (3.23) measures how incompatible model parameter vector ϕ is with the available data $\mathcal{D}_{1:t}$. In fact, by the derivations in the previous section: the mixture of variational distributions $\frac{1}{t}\sum_{\tau=1}^{t} q(\phi|\varphi_\tau)$ quantifies how likely a vector ϕ is on the basis of the data $\mathcal{D}_{1:t}$ (Fig. 3.5b); and the negative logarithm in (3.23) evaluates the information-theoretic "surprise" associated with that mixture. Problem (3.24) can be addressed either by grid search for low-dimensional model parameter space, or by using gradient ascent due to the differentiability nature of the scoring function (3.23), as illustrated in Fig. 3.5c.

In step *(ii)*, we need to convert the selected model parameter vector ϕ_{t+1}, i.e., the outcome of (3.24), into channel state c_{t+1}. We choose the channel state c_{t+1} that minimizes

the cross entropy loss when evaluated at ϕ_{t+1}, i.e.,

$$c_{t+1} \in \operatorname*{argmin}_{c} \Big\{ \mathcal{L}_{\mathrm{p}}(\phi_{t+1}|c) = \mathrm{E}_{(\mathbf{x},\mathbf{y})\sim p_0(x,y|c)}[-\log p(\mathbf{y}|\mathbf{x},\phi_{t+1})] \Big\}, \tag{3.25}$$

where we set $p_0(x,y|c) = p_0(y)p_0(x|y,c)$, with $p_0(y)$ being some fixed distribution and $p_0(x|y,c)$ being the distribution of the output of the simulator. In (3.25), we have emphasized that there may be more than one solution to the problem. The rational behind problem (3.25) is that data generated from the distribution $p_0(y,x|c_{t+1})$ can be interpreted as being the most compatible with the demodulator $p(y|x,\phi_{t+1})$, where compatibility is measured by the average of the cross entropy $\mathrm{E}_{\mathbf{x}\sim p(x|c_{t+1})}\Big[\mathrm{H}\Big(p_0(\mathbf{y}|\mathbf{x},c_{t+1}), p(\mathbf{y}|\mathbf{x},\phi_{t+1})\Big)\Big]$.

We emphasize that the proposed approach is different from the methodology introduced by [65], which uses another variational distribution in problem (3.20). In our experiments, we found the method in [65] to be ineffective and complex for the problem under study here. The main issue appears to be overfitting for the additional variational distribution, which is overcome by leveraging the availability of the channel simulator implementing the model $p_0(x|y,c)$.

In some models, problem (3.25) can be solved analytically. For more complex models, SGD-based approaches can be used, either by differentiating an estimate of the loss in a manner similar to the discussion in Sec. 3.2.2 $\Big($i.e., Algorithm 1 with $\mathcal{G}(\phi_{t+1}) = \mathcal{L}_{\mathrm{p}}(\phi_{t+1}|c)\Big)$, or by directly estimating its gradient [133].

Finally, in step *(iii)*, meta-training data set $\mathcal{D}_{t+1} = \{(x_{t+1}[i], y_{t+1}[i])\}_{i=1}^{N_{t+1}}$ is generated using the simulator in an i.i.d. fashion following the distribution

$$p(\mathcal{D}_{t+1}|c_{t+1}) = \prod_{i=1}^{N_{t+1}} p_0(y_{t+1}[i]) p_0\Big(x_{t+1}[i]\Big|y_{t+1}[i], c_{t+1}\Big). \tag{3.26}$$

As a final note, we adopt the proposal in [65] of implementing active selection only after $t_{\mathrm{init}} > 1$ channel states that are generated at random, as a means to avoid being overconfident at early stages. The overall proposed Bayesian active meta-learning scheme is summarized in Algorithm 3.

3.4 Experiments

In this section, we present experimental results to evaluate the performance of Bayesian meta-learning for demodulation/equalization.

Algorithm 3: `Bayesian Active Meta-Training`

Inputs : $p_0(x|y,c)$ = channel model
$p_0(y)$ = generative symbols distribution
Parameters : t_{init} = number of prior-based first frames
Output : ξ = shared hyperparameter vector

◁ *Generate initial experience* ▷
for $t = 1, 2, \ldots, t_{init}$ **do**
  Draw using the prior $c_t \sim p(c_t)$
  Acquire data $\mathcal{D}_t \sim \prod_{i=1}^{N_t} p_0(y_t[i]) p_0(x_t[i]|y_t[i], c_t)$
while *data acquisition not done* **do**
  $\xi \leftarrow$ `BayesianMetaTraining`$(\mathcal{D}_{1:t})$ using Algorithm 2
  ◁ *frame-specific update with updated* ξ ▷
  for $\tau = 1, 2, \ldots, t$ **do**
    $\varphi_\tau \leftarrow \varphi^{\text{GD}}(\mathcal{D}_\tau^{\text{tr}}|\xi)$ using (3.18)
  ◁ *step* (i)*, choose surprising model parameter* ▷
  $\phi_{t+1} = \text{argmax}_\phi \, s_t(\phi|\varphi_{1:t})$ using (3.23)
  ◁ *step* (ii)*, choose next channel* ▷
  $c_{t+1} \in \text{argmin}_c \, \mathcal{L}_{\text{p}}(\phi_{t+1}|c)$ as in (3.25)
  ◁ *step* (iii)*, generate data set* ▷
  Draw $\mathcal{D}_{t+1} \sim \prod_{i=1}^{N_{t+1}} p_0(y_{t+1}[i]) p_0(x_{t+1}[i]|y_{t+1}[i], c_{t+1})$
  $t \leftarrow t + 1$
return ξ

3.4.1 Performance Metrics

Apart from the standard measures of symbol error rate (SER) and mean squared error (MSE), we will also evaluate metrics quantifying the performance in terms of the reliability of the confidence measures provided by the predictor. While such measures can be defined for both classification and regression problems, we will focus here on uncertainty quantification for demodulation via calibration metrics (see [134] for discussion on regression).

As discussed in the previous sections, for a new frame, we need to make a prediction for the payload symbols $\{x_*^{\text{te}}[i]\}_{i=1}^{N_*^{\text{te}}}$ via the demodulator $p(y_*^{\text{te}}[i]|x_*^{\text{te}}[i], \phi_*)$ for frequentist meta-learning (2.31), or $p(y_*^{\text{te}}[i]|x_*^{\text{te}}[i], \varphi_*)$ for Bayesian meta-learning (3.19)

$$p(y_*^{\text{te}}[i]|x_*^{\text{te}}[i], \mathcal{D}_*^{\text{tr}}, \xi) = \begin{cases} p(y_*^{\text{te}}[i]|x_*^{\text{te}}[i], \phi^{\text{GD}}(\mathcal{D}_*^{\text{tr}}|\xi)), & \text{frequentist learning} \\ p(y_*^{\text{te}}[i]|x_*^{\text{te}}[i], \varphi_*(\mathcal{D}_*^{\text{tr}}|\xi)), & \text{Bayesian learning.} \end{cases} \tag{3.27}$$

The *confidence level* assigned by the model to the hard predicted symbol

$$\hat{y}_*^{\text{te}}[i] = \underset{y' \in \mathcal{Y}}{\text{argmax}}\, p(y'|x_*^{\text{te}}[i], \mathcal{D}_*^{\text{tr}}, \xi) \tag{3.28}$$

given the received symbol $x_*^{\text{te}}[i]$, can be defined as the corresponding probability [10]

$$\hat{p}[i] = \max_{y' \in \mathcal{Y}} p(y'|x_*^{\text{te}}[i], \theta) = p(\hat{y}_*^{\text{te}}[i]|x_*^{\text{te}}[i], \theta), \tag{3.29}$$

Perfect calibration [10] can be defined as the condition where symbols that are assigned a confidence level $\hat{p}[i]$ are also characterized by a probability of correct detection equal to p.

3.4.2 Frequentist and Bayesian Meta-Learning for Demodulation

For the first set of experiments, we focus on a demodulation problem at the symbol level in the presence of transmitter I/Q imbalance [135, 136], as considered also in [49]. The main reason for this choice is that channel decoding typically requires a hard decision on the transmitted codeword, whose accuracy can be validated via a cyclic redundancy check. In contrast, demodulation is usually a preliminary step at the receiver side, and downstream blocks, such as channel decoding, expect soft inputs that are well calibrated. For each frame τ, the transmitted symbols $y_\tau[i]$ are drawn uniformly at random from the 16-QAM constellation $\mathcal{Y} = 1/\sqrt{10}(\{\pm 1, \pm 3\} + \jmath\{\pm 1, \pm 3\})$. The received symbol $x_\tau[i] \in \mathcal{X} = \mathbb{C}$ is given as

$$x_\tau[i] = h_\tau f_{\text{IQ},\tau}(y_\tau[i]) + v_\tau[i], \tag{3.30}$$

for a unit energy fading channel coefficient h_τ, where the additive noise is $v_\tau[i] \sim \mathcal{CN}(0, \text{SNR}^{-1})$ for some signal-to-noise ratio (SNR) level SNR, and the I/Q imbalance function [137] $f_{\text{IQ},\tau} : \mathcal{Y} \to \bar{\mathcal{Y}}_\tau$ is

$$\begin{aligned} f_{\text{IQ},\tau}(y_\tau[i]) &= \bar{y}_{\text{I},\tau}[i] + \jmath \bar{y}_{\text{Q},\tau}[i] \\ \begin{bmatrix} \bar{y}_{\text{I},\tau}[i] \\ \bar{y}_{\text{Q},\tau}[i] \end{bmatrix} &= \begin{bmatrix} 1+\epsilon_\tau & 0 \\ 0 & 1-\epsilon_\tau \end{bmatrix} \begin{bmatrix} \cos\delta_\tau & -\sin\delta_\tau \\ -\sin\delta_\tau & \cos\delta_\tau \end{bmatrix} \begin{bmatrix} y_{\text{I},\tau}[i] \\ y_{\text{Q},\tau}[i] \end{bmatrix}, \end{aligned} \tag{3.31}$$

which depends on the imbalance parameters ϵ_τ and δ_τ. In (3.30), $y_{\text{I},\tau}[i]$ and $y_{\text{Q},\tau}[i]$ refer to the real and imaginary parts of the modulated symbol $y_\tau[i]$; and $\bar{y}_{\text{I},\tau}[i]$ and $\bar{y}_{\text{Q},\tau}[i]$ stand for the real and imaginary parts of the transmitted symbol $f_{\text{IQ},\tau}(y_\tau[i])$. Note that the constellation $\bar{\mathcal{Y}}_\tau$ of the transmitted symbols $\bar{y}_\tau[i]$ is also composed of 16 points via (3.31).

By (3.30) and (3.31), the channel state $\mathbf{c}_\tau$ consists of the tuple: (a) amplitude imbalance factor $\boldsymbol{\epsilon}_\tau \in [0, 0.15]$; (b) phase imbalance factor $\boldsymbol{\delta}_\tau \in [0, 15^\circ]$; and (c) channel realization $\mathbf{h}_\tau \in \mathbb{C}$. All of the variables are drawn i.i.d. across different frames and are fixed during each frame. We consider the channel state distribution for frame τ as

$$p_0(c_\tau) = \text{Beta}\left(\frac{\epsilon_\tau}{0.15}\middle|5,2\right)\text{Beta}\left(\frac{\delta_\tau}{0.15^\circ}\middle|5,2\right)\mathcal{CN}(h_\tau|0,1). \tag{3.32}$$

We set our base learner to be a multi-layer fully-connected neural network (3.3) with $L = 5$ layers. The real and imaginary parts of the input $x_\tau[i] \in \mathbb{C}$ are treated as a vector in $\mathbb{R}^2$, which is fed to layers with 10, 30, and 30 neurons, all with ReLU activations, while the last linear layer implements a softmax function that produces probabilities for the 16QAM constellation points.

To address the ability of meta-learning to adapt the demodulator using only few pilots, we set the number of pilots as $N_\tau^{\text{tr}} = 4$ during meta-training and $N_*^{\text{tr}} = 8$ for meta-testing [49]. Fig. 3.6 shows the SER as a function of the number of total meta-training frames t. Since only half of the constellation points are available as pilots during meta-test ($N_*^{\text{tr}} = 8$ different symbols out of 16), conventional learning cannot obtain a SER lower than of 0.5. In fact, conventional learning performs worse than a standard model-based receiver applying linear minimal mean square error (LMMSE), followed by maximum likelihood (ML) demodulation, while disregarding the presence of I/Q imbalance function f_{IQ}. Both meta-learning schemes are clearly superior to conventional learning and to the mentioned model-based solution, showing that useful knowledge has been transferred from previous frames to a new frame. Furthermore, Bayesian meta-learning obtains a slightly lower SER as compared to frequentist meta-learning. This advantage stems from the capacity of ensemble predictors to implement more complex decision boundaries [132].

To gain insights into the reliability of the uncertainty quantification provided by the demodulator, we use the metrics defined in Sec. 2.2, by setting the total number of bins which partition the $[0, 1]$ probability values to $M = 10$. We plot the ECE as a function of the number of total meta-training frames t in Fig. 3.7. Bayesian meta-learning is seen to achieve a lower ECE than frequentist meta-learning, indicating that Bayesian meta-learning provides more reliable estimates of uncertainty. Furthermore, the increase in ECE as the number t of available meta-training frames increases may be interpreted as a consequence of meta-overfitting [138]. This suggests that meta-learning may be considered as complete after a number of frames that depends on the complexity of the propagation environment. In practice, this can be assessed by evaluating the performance of the demodulator on pilots (see the online strategy in [49] for further discussion on this point). More meta-

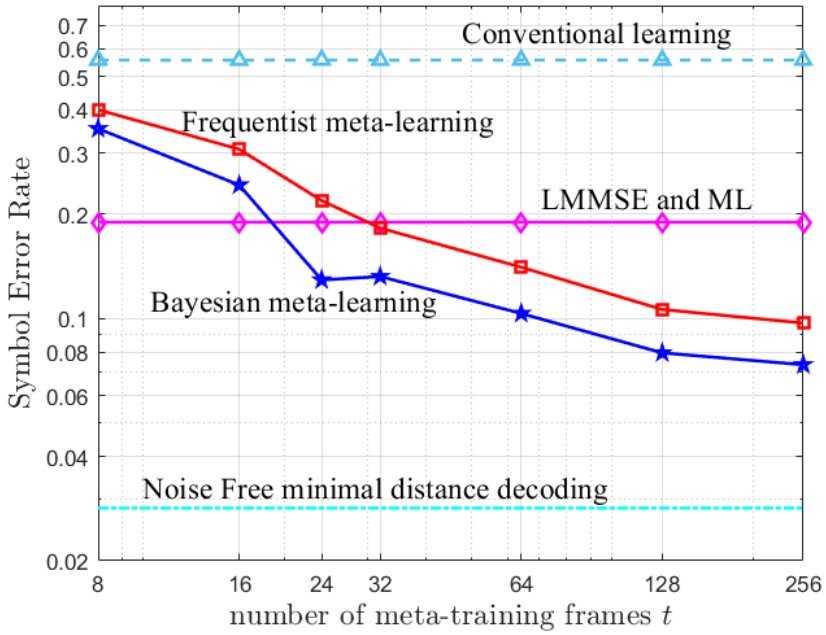


Fig. 3.6 Symbol error rate (SER) as a function of the number t of meta-training frames with 16-QAM, Rayleigh fading, and I/Q imbalance for $N_\tau^{\rm tr} = 4$, $N_*^{\rm tr} = 8$. The symbol error rate is averaged over by $N_*^{\rm te} = 4000$ data symbols and 50 meta-test frames with ensemble of size 100.

training tasks means higher diversity and complexity. If the meta-training tasks vary one from the other, the meta-learner may find it harder to capture the common patterns that generalize across tasks, and may turn unsuccessful to adapt effectively to the range of tasks encountered during meta-testing. Fine-tuning of the model architecture, choice of optimization algorithm, hyperparameters, and convergence criteria, can help to mitigate this effect, and is outside of the scope of this frequentist-Bayesian comparative work.

To further elaborate on the quality of uncertainty quantification, Fig. 3.8 depicts reliability diagrams for frequentist and Bayesian meta-learning. The within-bin accuracy levels $\mathrm{acc}(\mathcal{B}_m)$ in (2.11) and the within-bin empirical confidence $\mathrm{conf}(\mathcal{B}_m)$ in (2.12) are depicted as dark (blue) and light (red) bars, respectively. Frequentist meta-learning is observed to produce generally over-confident predictions, while Bayesian meta-learning provides better calibrated predictions with well-matching confidence and accuracy levels.

3.4.3 Bayesian Active Meta-Learning for Equalization

In this section, we illustrate the operation of active meta-learning by investigating a single-input multiple-output (SIMO) Rayleigh block fading real channel model. At frame τ,

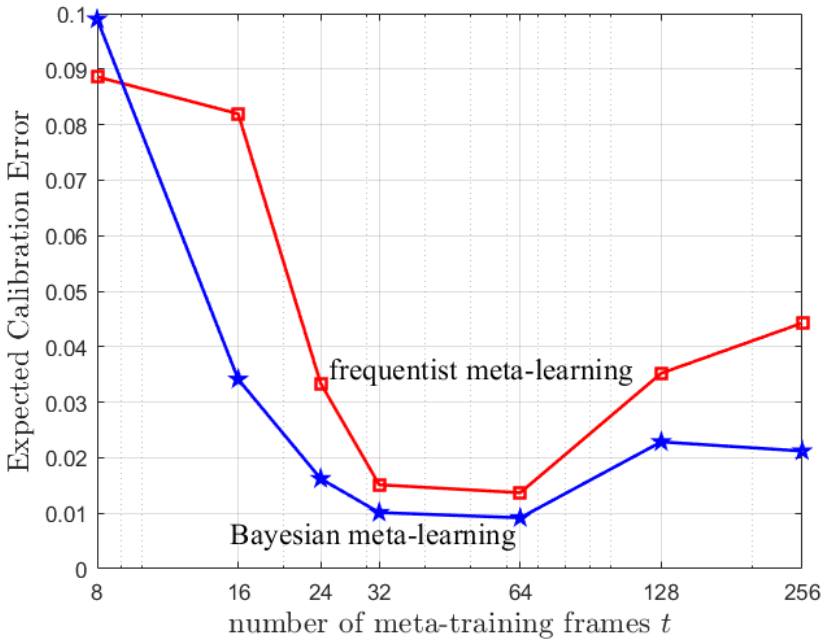


Fig. 3.7 Expected calibration error (ECE) over meta-test data $\mathcal{D}_*^{\text{te}}$ as a function of the number t of meta-training frames, for the same setting as in Fig. 3.6.

the modulator uses a 4-PAM to produce symbols $y_\tau[i]$, $i = 1, 2, \ldots, N_\tau$, taken uniformly from the set $\mathcal{Y} \in 1/\sqrt{5}\{-3, -1, +1, +3\}$. Given channel state $c_\tau = [c_\tau^0, c_\tau^1]^\top \in \mathbb{R}^2$, the i-th channel output symbol $x_\tau[i] \in \mathbb{R}^2$ for $i = 1, 2, \ldots, N_\tau$ is defined as the two-dimensional real vector

$$x_\tau[i] = c_\tau y_\tau[i] + v_\tau[i], \tag{3.33}$$

where both the additive noise $\mathbf{v}_\tau[i] \sim p_0(v) = \mathcal{N}\left(0, \frac{1}{2\text{SNR}} I_2\right)$ and the normalized real block fading coefficients $\mathbf{c}_\tau \sim p_0(c) = \mathcal{N}(c|0, I_2)$ are i.i.d. We adopt the linear equalizer

$$\hat{y}_\tau[i] = \phi_\tau^\top \cdot x_\tau[i] \tag{3.34}$$

with linear equalizer weight vector $\phi_\tau = [\phi_\tau^0, \phi_\tau^1]^\top \in \mathbb{R}^2$. To obtain a soft equalization, we account for a precision level β via the conditional distribution

$$p(y_\tau[i]|x_\tau[i], \phi_\tau) = \mathcal{N}(y_\tau[i]|\phi_\tau^\top \cdot x_\tau[i], \beta^{-1}). \tag{3.35}$$

The next model parameter ϕ_{t+1} is chosen to maximize the scoring function as in (3.24) by restricting the optimization to the domain $||\phi|| \leq 1$. This restricted optimization domain is selected in order to match the circular symmetry of the problem. Furthermore, the

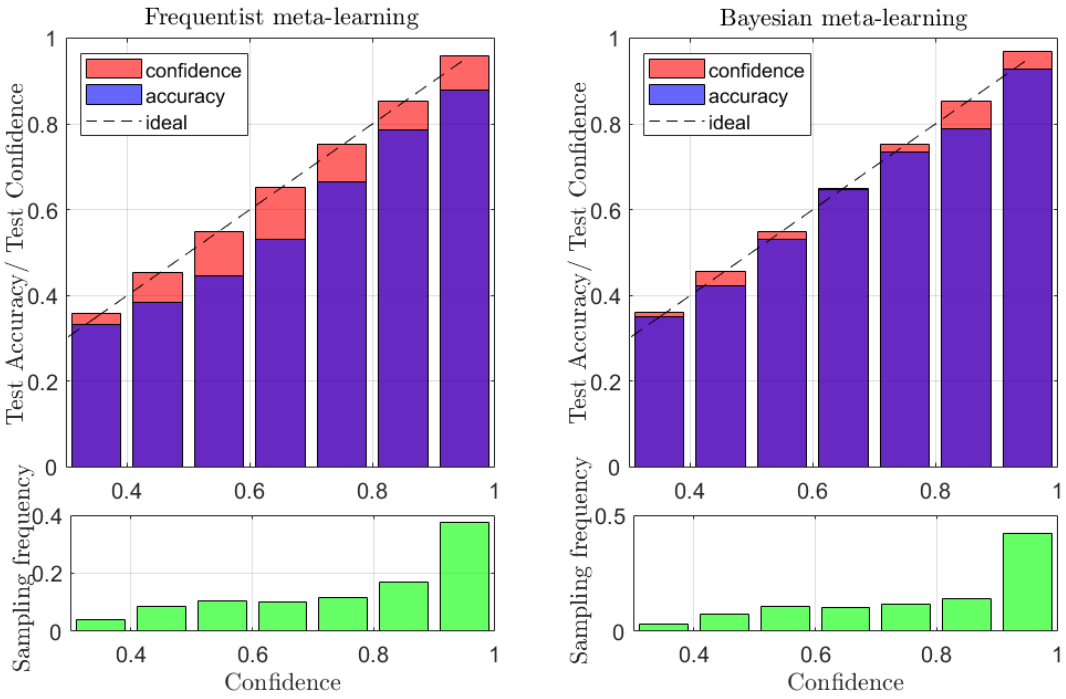


Fig. 3.8 Reliability diagrams (top) for frequentist meta-learning (left) and Bayesian meta-learning (right) with SNR = 18 dB, using $t = 16$ meta-training frames and predictions averaged over 50 meta-test frames. Frequentist meta-learning tends to be over-confident, whereas the Bayesian soft predictions are better matched to the true accuracy. The bottom figure shows the histogram of $|\mathcal{B}_m|/N$ of prediction over $M = 10$ bins. Full details in Appendix A.

corresponding next channel state c_{t+1} is selected by tackling problem (3.25), which amounts to the minimization

$$\begin{aligned} c_{t+1}(\phi) &\in \underset{c}{\operatorname{argmin}}\, \mathrm{E}_{(\mathbf{x},\mathbf{y})\sim p_0(y)p_0(x|y,c)}[-\log p(\mathbf{y}|\mathbf{x},\phi)] && \text{(3.36a)} \\ &= \underset{c}{\operatorname{argmin}}\, \mathrm{E}_{(\mathbf{y},\mathbf{v})\sim p_0(y)p_0(v)}[\tfrac{\beta}{2}(\mathbf{y}-\phi^\top\cdot(c\mathbf{y}+\mathbf{v}))^2] \\ &= \underset{c}{\operatorname{argmin}}\, \mathrm{E}_{(\mathbf{y},\mathbf{v})\sim p_0(y)p_0(v)}[\tfrac{\beta}{2}\Big((1-\phi^\top\cdot c)\mathbf{y}-\phi^\top\mathbf{v}\Big)^2] \\ &= \Big\{c\Big|\phi^\top\cdot c=1\Big\}. && \text{(3.36b)} \end{aligned}$$

In the set of solutions of problem (3.36b), we select the minimum-norm solution $c_{t+1} = \phi_{t+1}/\|\phi_{t+1}\|^2$. This way, the selected channel focuses on the more challenging low-SNR regime. Details of this experiment are provided in Appendix A.

Fig. 3.9 illustrates the scoring function (3.23) used to select the next model parameter ϕ_{t+1} as a heat map in the space of model parameter ϕ. Specifically, the figure shows the scoring functions after observing $t = 4$ and $t = 5$ meta-training frames. The optimized next model parameter vector ϕ_{t+1} (3.24) is shown as a star, while the previously selected model parameter vectors $\phi_{1:t}$ are shown as squares. Fig. 3.9 illustrates how active meta-learning efficiently explores the model parameter space. It does so by avoiding the inclusion of channel states that are similar to those already considered (i.e., the squares in the figure). This way, the model parameter space can be covered with fewer meta-training frames t, leading to a larger frame efficiency of active meta-learning.

Finally, to numerically validate the advantage of active meta-learning, we plot the meta-test MSE loss in Fig. 3.10 for both passive and active Bayesian meta-learning versus the number of frames t. For passive meta-learning, we have generated random channel realizations by drawing from the distribution $p_0(c) = \mathcal{N}(c|0, I_2)$. We have repeated the experiment 100 times, and show the confidence interval of one standard deviation for the meta-test loss. The results in the figure confirm that active meta-learning requires far fewer meta-training frames. Furthermore, the increased randomness of passive meta-learning is due to the random selection of channel states at each iteration.

3.5 Related Work

Bayesian learning has been applied to communication systems in various works. The work [139] applied it to the problem of predicting the number of active users in LTE system; papers [140, 141] applied MC-based Bayesian learning for MIMO detection; the works [142–

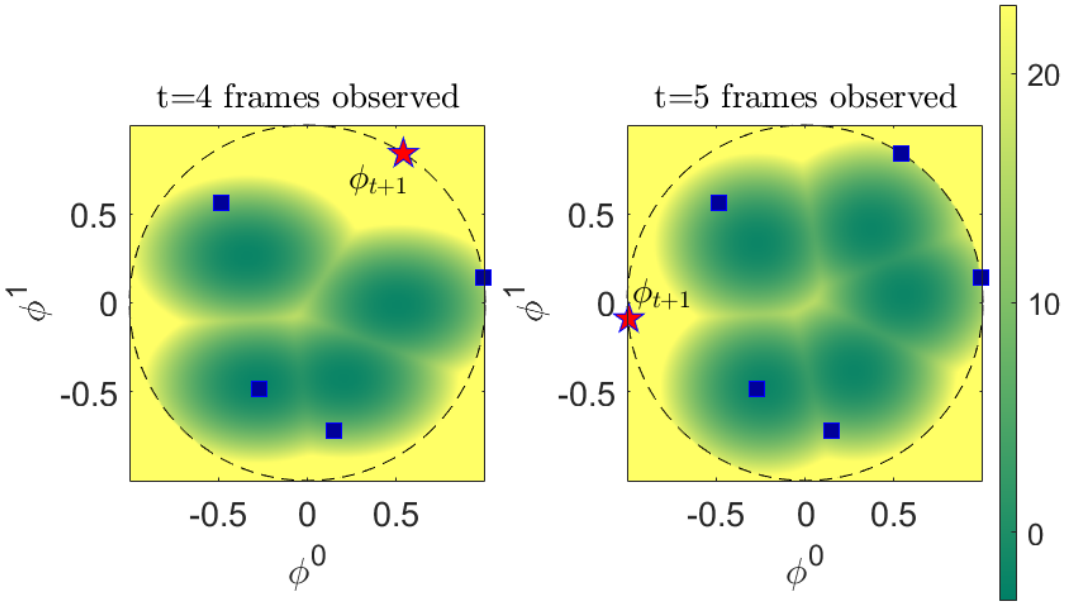


Fig. 3.9 Scoring function (3.23) used by Bayesian active meta-learning to select the next model parameter vector ϕ_{t+1} at the fourth and fifth iterations. The scoring function is shown as a heat map over the two dimensional space of the model parameter vector ϕ for the example detailed in Sec. 3.4.3.

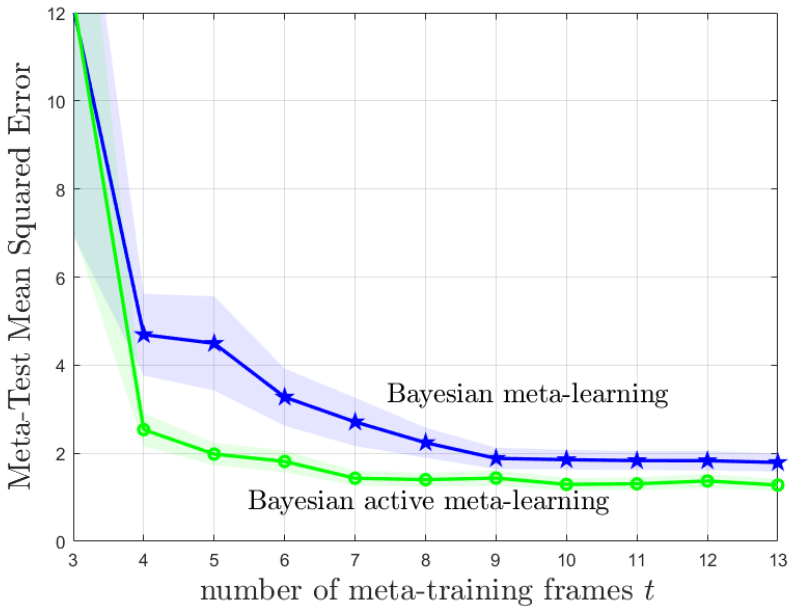


Fig. 3.10 Meta-test mean squared error (MSE) loss as function of the number of frames t. Bayesian active meta-training is able to achieve lower meta-test loss levels by using fewer meta-training tasks t. Solid lines are the mean test loss over 100 channel states. The confidence levels account for one standard deviation across the experiments.

144] addressed channel prediction/estimation for massive MIMO systems; reference [145] studied the identification of IoT transmitters; multi user detection was applied in [107, 108]; and the authors of [132] proposed the use of robust Bayesian learning for modulation classification, localization, and channel modeling.

As for active learning, applications to communication systems include paper [146], which proposed a sample-efficient retransmission protocol; reference [147], which tackled initial beam alignment for massive MIMO system; work [148], which aimed at mitigating the problem of scarce training data in wireless cyber-security attack; and reference [149], which addressed resource allocation problems in vehicular communication systems.

Like Bayesian learning, meta-learning also provides a general design principle, which can be implemented by following different approaches. Optimization-based methods design the hyperparameters used by training algorithms; model-based techniques optimize an additional neural network model to guide adaptation of the main AI model; and metric-based schemes identify metric spaces for non-parametric inference (see, e.g., [105] and references therein).

The integration of meta-learning and Bayesian learning is highly non-trivial, and is an active topic of research in the machine learning literature. References [150, 30, 151–153] addressed Bayesian meta-learning via empirical Bayes using parametric VI [150, 30], particle-based VI [151], deep-kernels [152], and expectation-maximization [153]; while the papers [123, 154–156] studied full Bayesian meta-learning that treats also the hyperparameters as random variables. Lastly, the work [157] proposed the use of quantum machine learning models as parameterized variational distributions.

3.6 Conclusions

In this chapter, we have introduced tools for reliable and efficient AI in communication systems via Bayesian meta-learning. Bayesian learning has the advantage of producing well-calibrated decisions whose confidence levels are a close match for the corresponding test accuracy. This property facilitates monitoring of the quality of the outputs of an AI module. Meta-learning optimizes models that can quickly adapt based on few pilots, producing sample-efficient AI solutions. This paper has focused on the application of Bayesian meta-learning to the basic problems of demodulation/equalization from few pilots. We have demonstrated via experiments that the demodulator/equalizer obtained via Bayesian meta-learning not only achieves a higher accuracy, but it also enjoys better calibration performance than its standard frequentist counterpart. Furthermore, thanks to

meta-learning, such performance levels can be obtained based on a limited number of pilots per frame.

To reduce the number of past frames required by meta-learning, we have also introduced Bayesian active meta-learning, which leverages the uncertainty estimates produced by Bayesian learning to actively explore the space of channel conditions. We have shown via numerical results that active meta-learning can indeed significantly speed up meta-training in terms of number of frames.

Chapter 4

Calibrating AI Models via Conformal Prediction

4.1 Introduction

4.1.1 Conformal Prediction for AI-based Wireless Systems

CP leverages probabilistic predictors to construct well-calibrated *set predictors*. Instead of producing a probability vector, as in the examples in Fig. 2.1, a set predictor outputs a subset of the output space, as exemplified in Fig. 4.1. A set predictor is *well calibrated* if it contains the correct output with a pre-defined *coverage* probability selected by the system designer. For a well-calibrated set predictor, the size of the prediction set for a given input provides a measure of the uncertainty of the decision. Set predictors with smaller average prediction size are said to be more *efficient* [17].

This chapter investigates CP as a general mechanism to obtain AI models with formal calibration guarantees for communication systems. The calibration guarantees of CP hold irrespective of the true, unknown, distribution underlying the generation of the variables of interest, and are defined either in terms of ensemble averages [17] or in terms of long-term averages [83]. CP is applied in conjunction to both frequentist and Bayesian learning, and specific applications are discussed to demodulation, modulation classification, and channel prediction.

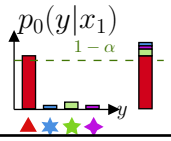
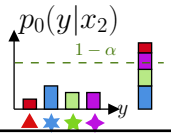
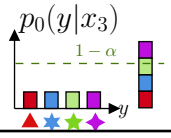
input	ground-truth distribution	well-calibrated efficient	well-calibrated inefficient	poorly-calibrated efficient
x_1		{▲ }	{▲★ }	{✦ }
x_2		{★✦ }	{▲★✦ }	{★★ }
x_3		{▲★★ }	{▲★★✦ }	{★★ }

Fig. 4.1 Set predictors produce subsets of the range of the output variable (here { ▲★★✦ }) for each input. Calibration is measured with respect to a desired coverage level $1-\alpha$: A set predictor is well calibrated if the true label is included in the prediction set with probability at least $1-\alpha$. A well-calibrated set predictor can be inefficient if it returns excessively large set predictions (forth column). In contrast, a poorly-calibrated set predictor (fifth column) returns set predictions that include the true value of the label with a probability smaller than $1-\alpha$.

4.2 Data-Generation Model

We consider the standard supervised learning setting in which the learner is given a data set $\mathcal{D} = \{z[i]\}_{i=1}^N$ of N examples of input-output pairs $z[i] = (x[i], y[i])$ for $i = 1, \ldots, N$, and is tasked with producing a prediction on a test input x with unknown output y. Writing $z = (x, y)$ for the test pair, data set $\mathcal{D}$ and test point z follow the unknown *ground-truth*, or *population*, *distribution* $p_0(\mathcal{D}, z)$. Apart from Sec. 4.5, we further assume throughout that the population distribution $p_0(\mathcal{D}, z)$ is *exchangeable* – a condition that includes as a special case the traditional independent and identically distributed (i.i.d.) data-generation setting.

Mathematically, exchangeability requires that the joint distribution $p_0(\mathcal{D}, z)$ does not depend on the ordering of the $N+1$ variables $\{z[1], \ldots, z[N], z\}$. Equivalently, by de Finetti's theorem [158], there exists a latent random vector $\mathbf{c}$ with distribution $p_0(c)$ such that, conditioned on $\mathbf{c}$, the variables $\{z[1], \ldots, z[N], z\}$ are i.i.d. Writing the conditional i.i.d. distribution as

$$p_0(\mathcal{D}, z|\mathbf{c}) = p_0(z|\mathbf{c}) \prod_{i=1}^{N} p_0\Big(z[i]\Big|\mathbf{c}\Big) \tag{4.1}$$

for some ground-truth sampling distribution $p_0(z|c)$ given the variable c, under the exchangeability assumption, the joint distribution can be expressed as

$$p_0(\mathcal{D}, z) = \mathrm{E}_{\mathbf{c}\sim p_0(c)}\Big[p_0(\mathcal{D}, z|\mathbf{c})\Big]. \tag{4.2}$$

The vector $\mathbf{c}$ in (4.2) can be interpreted as including context variables that determine the specific learning task. For instance, in a wireless communication setting, the vector $\mathbf{c}$ may encode information about channel conditions such as distortion and channel gain, while the data set $\mathcal{D}$ and point z account for payload symbols which are distorted and noised in an identical and independent manner when facing the same channel conditions within a transmission packet. This setting is investigated in Sec. 4.7. In Sec. 4.5, we will consider a more general setting in which no assumptions are made on the distribution of the data.

4.3 Naïve Set Predictors

Before describing CP in the next section, in this section we review two naïve , but natural and commonly used, approaches to produce set predictors, that fail to satisfy the coverage condition (2.32).

4.3.1 Naïve Set Predictors from Probabilistic Predictors

Given a probabilistic predictor $p(y|x, \mathcal{D})$ as in (2.1), one could construct a set predictor by relying on the confidence levels reported by the model. Specifically, aiming at satisfying the coverage condition (2.32), given an input x, one could construct the smallest subset of the output domain $\mathcal{Y}$ that covers a fraction $1-\alpha$ of the probability designed by model $p(y|x, \mathcal{D})$. Mathematically, the resulting *naïve probabilistic-based* (NPB) set predictor is defined as

$$\Gamma^{\mathrm{NPB}}(x|\mathcal{D}) = \underset{\Gamma\in 2^{\mathcal{Y}}}{\operatorname{argmin}} |\Gamma| \text{ s.t. } \sum_{y'\in\Gamma} p(y'|x, \mathcal{D}) \geq 1-\alpha \tag{4.3}$$

for the case of a discrete set, and an analogous definition applies in the case of a continuous domain $\mathcal{Y}$. Fig. 4.2 illustrates the NPB for a prediction problem with output domain size $|\mathcal{Y}| = 4$. Given that, as mentioned in Sec. 4.1, probabilistic predictors are typically poorly calibrated, the naïve set predictor (4.3) does not satisfy condition (2.32) for the given desired miscoverage level α, and hence it is not well calibrated. For example, in the typical

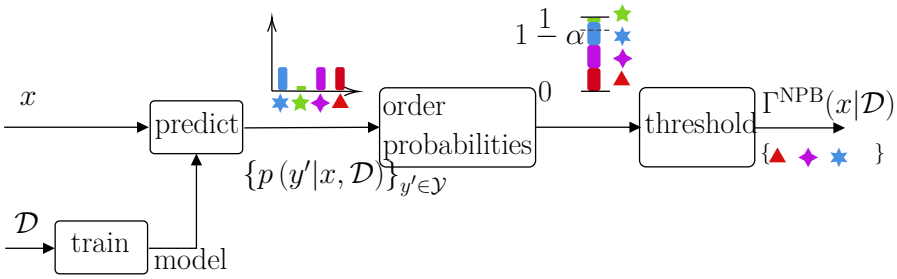


Fig. 4.2 A naïve probabilistic-based (NPB) set predictor uses a pre-trained probabilistic predictor to include all output values to which the probabilistic predictor assigns the largest probabilities that reach the coverage target $1-\alpha$. This naïve scheme has no formal guarantee of calibration, i.e., it does not guarantee the coverage condition (2.32), unless the original probabilistic predictor is well calibrated.

case in which the probabilistic predictor is overconfident [10], the predicted sets (4.3) tend to be too small to satisfy the coverage condition (2.32).

4.3.2 Naïve Set Predictors from Quantile Predictors

While the naïve probabilistic-based set predictor (4.3) applies to both discrete and continuous target variables, we now focus on the important special case in which $\mathcal{Y}$ is a real number, i.e., $\mathcal{Y}=\mathbb{R}$. This corresponds to scalar regression problems, such as for channel prediction (see Sec. 4.9). Under this assumption, one can construct a naïve set predictor based on estimates of the $\alpha/2$- and $(1-\alpha/2)$-quantiles $y_{\alpha/2}(x)$ and $y_{1-\alpha/2}(x)$ of the ground-truth distribution $p_0(y|x)$ (obtained from the joint distribution $p_0(\mathcal{D},z)$). In fact, writing as

$$y_q(x)=\inf\Big\{y\in\mathbb{R}:\int_{-\infty}^{y}p_0(y'|x)\,\mathrm{d}y'\leq q\Big\} \tag{4.4}$$

the q-quantile, with $q\in[0,1]$, of the ground-truth distribution $p_0(y|x)$, the interval $\big[y_{\alpha/2}(x),y_{1-\alpha/2}(x)\big]$ contains the true value y with probability $1-\alpha$.

Defining the pinball loss as [159]

$$\ell_q(y,\hat{y})=\max\Big\{-(1-q)(y-\hat{y}),q(y-\hat{y})\Big\} \tag{4.5}$$

for $q\in[0,1]$, the quantile $y_q(x)$ in (4.4) can be obtained as [160]

$$y_q(x)=\operatorname*{argmin}_{\hat{y}\in\mathbb{R}}\mathrm{E}_{\mathbf{y}\sim p_0(y|x)}\big[\ell_q(\mathbf{y},\hat{y})\big]. \tag{4.6}$$

Therefore, given a parametrized predictive model $\hat{y}(x|\phi)$, the quantile $y_q(x)$ can be estimated as $\hat{y}(x|\phi_{\mathcal{D},q})$ with optimized parameter vector

$$\phi_{\mathcal{D},q} = \underset{\phi}{\operatorname{argmin}} \left\{ \tfrac{1}{N} \sum_{(x,y)\in\mathcal{D}} \ell_q\big(y, \hat{y}(x|\phi)\big) \right\}. \tag{4.7}$$

With the estimate $\hat{y}(x|\phi_{\mathcal{D},\alpha/2})$ of quantile $y_{\alpha/2}(x)$ and estimate $\hat{y}(x|\phi_{\mathcal{D},1-\alpha/2})$ of quantile $y_{1-\alpha/2}(x)$, we finally obtain the *naïve quantile-based* (NQB) predictor

$$\Gamma^{\text{NQB}}(x|\mathcal{D}) = \Big[\hat{y}(x|\phi_{\mathcal{D},\alpha/2}), \hat{y}(x|\phi_{\mathcal{D},1-\alpha/2})\Big]. \tag{4.8}$$

The naïve set prediction in (4.8) fails to satisfy the condition (2.32), since the empirical quantiles $\hat{y}_q(x)$ generally differ from the ground-truth quantiles $y_q(x)$.

4.4 Conformal Prediction

In this section, we review CP-based set predictors, which have the key property of guaranteeing the $(1-\alpha)$-validity condition (2.32) for any predetermined miscoverage level α, irrespective of the ground-truth distribution $p_0(\mathcal{D}, z)$ of the data. We specifically focus on validation-based CP [17] and cross-validation-based CP [90], which are more practical variants of full CP [17, 161]. For completeness, two full conformal predictions are presented in Appendix B.1. In Sec. 4.5, we cover online CP [83, 84].

4.4.1 Validation-based CP (VB-CP)

In this subsection, we describe *validation-based* CP (VB-CP), which partitions the available set $\mathcal{D} = \mathcal{D}^{\text{tr}} \cup \mathcal{D}^{\text{val}}$ into a training set $\mathcal{D}^{\text{tr}}$ with N^{tr} samples and a validation set $\mathcal{D}^{\text{val}}$ with $N^{\text{val}} = N - N^{\text{tr}}$ samples (Fig. 4.3(a)). This class of methods is also known as inductive CP [17] or split CP [90].

VB-CP operates on any pre-trained probabilistic model $p(y|x, \mathcal{D}^{\text{tr}})$ obtained using the training set $\mathcal{D}^{\text{tr}}$ as per (2.1). At test time, given an input x, VB-CP relies on a validation set to determine which labels $y' \in \mathcal{Y}$ should be included in the predicted set. Specifically, for any given test input x, a label $y' \in \mathcal{Y}$ is included in set $\Gamma^{\text{VB}}(x|\mathcal{D})$ depending on the extent to which the candidate pair (x, y') "conforms" with the examples in the validation set.

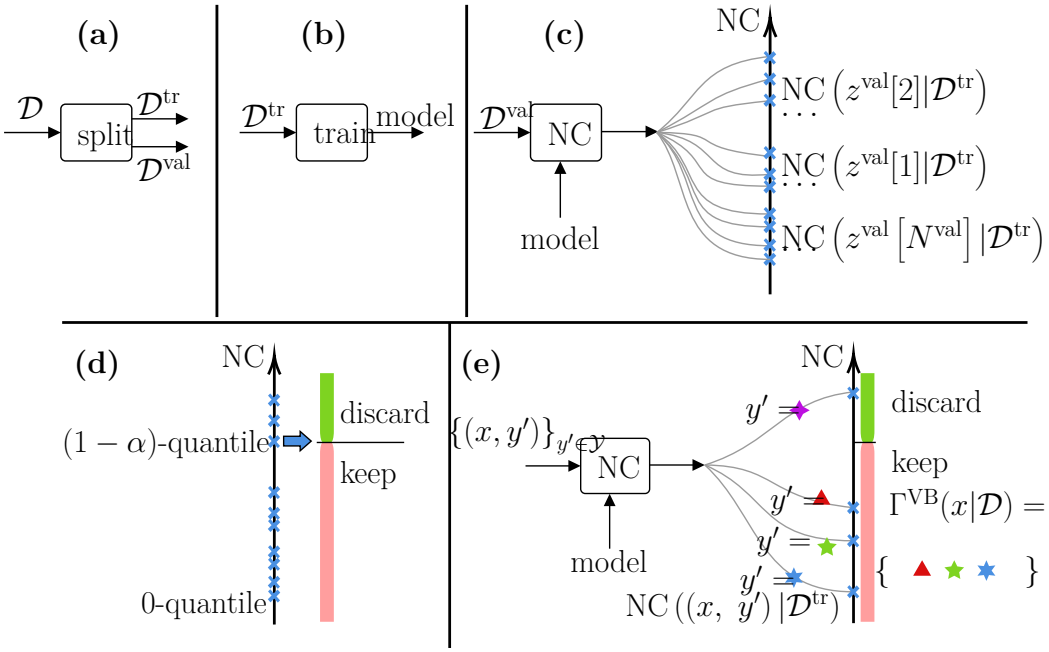


Fig. 4.3 Validation-based conformal prediction (VB-CP): **(a)** The data set is split into training and validation set; **(b)** A single model is trained over the training data set; **(c)-(d)** Post-hoc calibration is done by evaluating the NC scores on the validation set **(c)** and by identifying the $(1-\alpha)$-quantile of the validation NC scores. This divides the axis of NC scores into a "keep" region of NC scores smaller than the threshold, and into a complementary "discard" region **(d)**. **(e)** For each test input x, VB-CP includes in the prediction set all labels $y' \in \mathcal{Y}$ for which the NC score of the pair (x, y') is within the "keep" region.

This "conformity" test for a candidate pair is based on a *nonconformity (NC) score*. An NC score for VB-CP can be obtained as the log-loss

$$\text{NC}(z=(x,y)|\mathcal{D}^{\text{tr}}) = -\log p(y|x,\mathcal{D}^{\text{tr}}) \tag{4.9}$$

or as any other score function that measures the loss of the probabilistic predictor $p(y|x,\mathcal{D}^{\text{tr}})$ on example (x,y). It is also possible to define NC scores for quantile-based predictors as in (4.8), and we refer to [84] for details.

VB-CP consists of a training phase (Fig. 4.3(a)-(d)) and of a test phase (Fig. 4.3(e)). During *training*, the data set $\mathcal{D}^{\text{tr}}$ is used to obtain a probabilistic predictor $p(y|x,\mathcal{D}^{\text{tr}})$ as in (2.1) (Fig. 4.3(b)). Then, NC scores $\text{NC}(z^{\text{val}}[i]|\mathcal{D}^{\text{tr}})$, as in (4.9), are evaluated on all points $z^{\text{val}}[i], i=1,\ldots,N^{\text{val}}$ in the validation set $\mathcal{D}^{\text{val}}$ (Fig. 4.3(c)). Finally, the real line of NC scores is partitioned into a "keep" region and a "discard" region (Fig. 4.3(d)), choosing

as a threshold the $(1-\alpha)$-empirical quantile of the N^{val} NC scores $\{\text{NC}(z^{\text{val}}[i]|\mathcal{D}^{\text{tr}})\}_{i=1}^{N^{\text{val}}}$. Accordingly, we "keep" the labels y' with NC scores that are smaller than the $(1-\alpha)$-empirical quantile of the validation NC scores, and "discard" larger NC scores.

During *testing* (Fig. 4.3(e)), given a test input x, $|\mathcal{Y}|$ NC scores are evaluated, one for each of the candidate labels $y' \in \mathcal{Y}$, using the same trained model $p(y|x, \mathcal{D}^{\text{tr}})$. All candidate labels y' for which the NC score $\text{NC}((x, y')|\mathcal{D}^{\text{tr}})$ falls within the "keep" region are included in the predicted set of VB-CP.

Mathematically, the VB-CP set predictor is obtained as

$$\Gamma^{\text{VB}}(x|\mathcal{D}) = \Big\{ y' \in \mathcal{Y} \Big| \text{NC}((x, y')|\mathcal{D}^{\text{tr}}) \leq Q_\alpha\Big(\{\text{NC}(z^{\text{val}}[i]|\mathcal{D}^{\text{tr}})\}_{i=1}^{N^{\text{val}}}\Big) \Big\}, \tag{4.10}$$

where the *empirical quantile from the top* for a set of N real values $\{r[i]\}_{i=1}^{N}$ is defined as

$$Q_\alpha\left(\{r[i]\}_{i=1}^{N}\right) = \Big\lceil (1-\alpha)(N+1) \Big\rceil \text{th smallest value of the set } \{r[i]\}_{i=1}^{N} \cup \{+\infty\}. \tag{4.11}$$

4.4.2 Cross-validation-based CP (CV-CP)

VB-CP has the computational advantage of requiring the training of a single model, but the split into training and validation data causes the available data to be used in an inefficient way. This data inefficiency generally yields set predictors with a large average size (2.33). Unlike VB-CP, *cross-validation-based* CP (CV-CP) [90] trains multiple models, each using a subset of the available data set $\mathcal{D}$. As detailed next and summarized in Fig. 4.4, during the training phase, each data point $z[i]$ in the validation set is assigned an NC score based on a model trained using a subset of the data set $\mathcal{D}$ that excludes $z[i]$, with $i \in \{1, ..., N\}$. Then, for testing, the inclusion of a label y' in the prediction set for an input x is based on a comparison of NC scores evaluated for the pair (x, y') with all the N validation NC scores.

Specifically, as illustrated in Fig. 4.4, K-fold CV-CP [90], referred here as K-CV-CP, first partitions the data set $\mathcal{D}$ into K disjoint folds $\{\mathcal{S}_k\}_{k=1}^{K}$, each with N/K points, i.e., $\cup_{k=1}^{K} \mathcal{S}_k = \mathcal{D}$ (Fig. 4.4(a)), for a predefined integer $K \in \{2, \ldots, N\}$ such that the ratio N/K is an integer.

During *training*, the K subsets $\mathcal{D} \setminus \mathcal{S}_k$ are used to train K probabilistic predictors $p(y|x, \mathcal{D} \setminus \mathcal{S}_k)$ defined as in (2.1) (Fig. 4.4(b)). Each trained model $p(y|x, \mathcal{D} \setminus \mathcal{S}_k)$ is used to evaluate the $|\mathcal{S}_k| = N/K$ NC scores $\text{NC}\Big(z_k \Big| \mathcal{D} \setminus \mathcal{S}_k\Big)$ for all validation data points $z_k \in \mathcal{S}_k$ that were not used for training the model (Fig. 4.4(c)). Unlike VB-CP, K-CV-CP requires

keeping in memory all the N validation scores for testing. These points are illustrated as crosses in Fig. 4.4(c).

During *testing*, for a given test input x and for any candidate label $y' \in \mathcal{Y}$, CV-CP evaluates K NC scores, one for each of the K trained models. Each such NC score $\text{NC}\Big((x,y')\Big|\mathcal{D}\setminus\mathcal{S}_k\Big)$ is compared with the N/K validation scores obtained on fold $\mathcal{S}_k$. We then count how many of the N/K validation scores are larger than $\text{NC}\Big((x,y')\Big|\mathcal{D}\setminus\mathcal{S}_k\Big)$. If the sum of all such counts, across the K folds $\{\mathcal{S}_k\}_{k=1}^K$, is larger than a fraction α of all N data points, then the candidate label y' is included in the prediction set (Fig. 4.4(d)). This criterion follows the same principle of VB-CP of including all candidate labels y' that "conform" well with a sufficiently large fraction of validation points.

Mathematically, K-CV-CP is defined as

$$\Gamma^{K\text{-CV}}(x|\mathcal{D}) = \left\{ y' \in \mathcal{Y} \middle| \sum_{k=1}^{K} \sum_{z_k \in \mathcal{S}_k} \mathbb{1}\Big(\text{NC}\Big((x,y')\Big|\mathcal{D}\setminus\mathcal{S}_k\}\Big) \leq \text{NC}\Big(z_k\Big|\mathcal{D}\setminus\mathcal{S}_k\Big)\Big) \geq \lfloor \alpha(N+1) \rfloor \right\}, \tag{4.12}$$

where $\mathbb{1}(\cdot)$ is the indicator function ($\mathbb{1}(\text{true}) = 1$ and $\mathbb{1}(\text{false}) = 0$). The left-hand side of the inequality in (4.12) implements the sums, shown in Fig. 4.4(d), over counts of validation NC scores that are larger than the corresponding NC score for the candidate pair (x, y').

K-CV-CP increases the computational complexity K-fold as compared to VB-CP, while generally reducing the inefficiency [90]. The special case of $K = N$, known as jackknife+ [90], is referred here as CV-CP. In this case, each of the N folds $\mathcal{S}_k, k = 1, \ldots, N$ uses a single cross validation point. In general, CV-CP is the most efficient form of K-CV-CP, but it may be impractical for large data set sizes due to need to train N models. The number of folds K should strike a balance between computational complexity, as K models are trained, and inefficiency.

4.4.3 Calibration Guarantees

VB-CP (4.10) satisfies the coverage condition (2.32) [17] under the only assumption of exchangeability (see Sec. II-A).

The validity of CV-CP requires a technical assumption on the NC score. While in VB-CP the NC score is an arbitrary score function evaluated based on any pre-trained probabilistic model, for CV-CP, the NC score $\text{NC}(z|\mathcal{D}')$ must satisfy the additional property of being invariant to permutations of the data set $\mathcal{D}'$ used to train the underlying probabilistic model. Consider the log-loss $\text{NC}(z = (x,y)|\mathcal{D}') = -\log p(y|x,\mathcal{D}')$ (4.9), or any other score function based on the trained model $p(y|x,\mathcal{D}')$, as the NC score. CV-CP requires that the

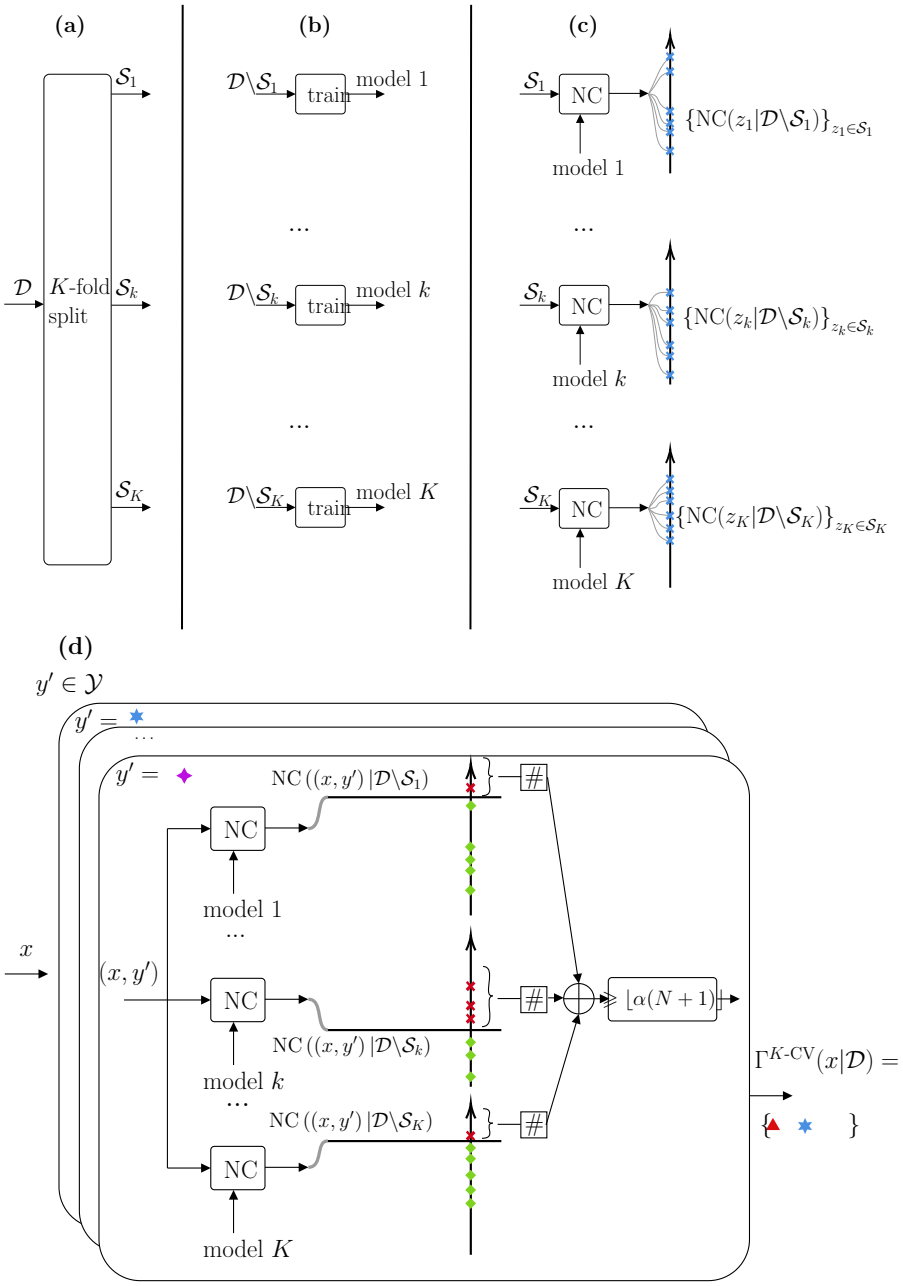


Fig. 4.4 K-fold cross-validation-based conformal prediction (K-CV-CP): **(a)** The N data pairs of data set $\mathcal{D}$ are split into K-folds each with $|\mathcal{S}_k| = N/K$ samples; **(b)** K models are trained, each using a leave-fold-out data set of $|\mathcal{D} \setminus \mathcal{S}_k| = N - N/K$ pairs; **(c)** NC scores are computed on the N/K holdout data points for each fold $\mathcal{S}_k$; **(d)** For each test input x, all labels $y' \in \mathcal{Y}$ for which the number of "higher-NC" validation points exceeds a fraction α of the total N points are considered in the prediction set. CV-CP is the special case with $K = N$.

training algorithm used to produce model $p(y|x, \mathcal{D}')$ provides outputs that are invariant to permutations of the training set $\mathcal{D}'$.

Specifically, for frequentist learning, the optimization algorithm producing the parameter vector $\phi^*_{\mathcal{D}'}$ in (2.1) must be permutation-invariant. This is the case for standard methods such as full-batch gradient descent (GD), or for non-parametric techniques such as Gaussian processes. For Bayesian learning, the distribution $q^*(\phi|\mathcal{D}')$ in (2.1) must also be permutation-invariant, which is true for the exact posterior distribution [28], as well as for approximations obtained via MC methods such as Langevin MC [39, 28].

The requirement on permutation-invariance can be alleviated by allowing for probabilistic training algorithms such as stochastic gradient descent (SGD) [162]. With probabilistic training algorithms, the only requirement is that the *distribution* of the (random) output models is permutation-invariant. This is, for instance, the case if SGD is implemented by taking mini-batches uniformly at random within the training set $\mathcal{D}'$ [162–164]. With probabilistic training algorithms, however, the validity condition (2.32) of CV-CP is only guaranteed on average with respect to the random outputs of the algorithms.

Specifically, under the discussed assumption of permutation-invariance of the NC scores, by [90, Theorems 1 and 4], CV-CP satisfies the inequality

$$\mathbb{P}\Big(\mathbf{y} \in \Gamma^{\text{CV}}(\mathbf{x}|\boldsymbol{\mathcal{D}})\Big) \geq 1 - 2\alpha, \tag{4.13}$$

while K-CV-CP satisfies the inequality

$$\mathbb{P}\Big(\mathbf{y} \in \Gamma^{K\text{-CV}}(\mathbf{x}|\boldsymbol{\mathcal{D}})\Big) \geq 1 - 2\alpha - \min\left\{\tfrac{2(1-1/K)}{N/K+1}, \tfrac{1-K/N}{K+1}\right\} \tag{4.14}$$

$$\geq 1 - 2\alpha - \sqrt{2/N}. \tag{4.15}$$

Therefore, validity for both cross-validation schemes is guaranteed for the larger miscoverage level of 2α. Accordingly, one can achieve miscoverage level of α, satisfying (2.32), by considering the CV-CP set predictor $\Gamma^{\text{CV}}(x|\mathcal{D})$ with $\alpha/2$ in lieu of α in (4.12). That said, in the experiments, we will follow the recommendation in [90] and [163] to use α in (4.12).

4.4.4 Complexity and Validity Guarantees of Set Predictors

We now focus on the complexity and validity guarantees involved with the set predictors reviewed in Section 4.4.1 and Section 4.4.2, along with Full CP which is given in detail in Appendix B.1. Recall that N is the data set size, and N^{te} stands for the total number of prediction points (x, y). Table 4.1 shows the complexity of the different methods, in terms

of the number of models being trained (a costly procedure); the number of predictions using these models (a less costly procedure); and the formal guaranteed level. All complexity values are associated with respect to a baseline of a single model training or prediction, meaning that Bayesian learning using MC ensembling will include a multiplicative term of the ensemble size. A typical method for regression problem is to consider candidate labels on a grid, hence the number of model predictions in Table 4.1 uses grid size instead of label cardinality $|\mathcal{Y}|$.

set predictor	# models trained	# model predictions	guarantees
naïve (4.3)	1	N^{te}	none
VB (4.10)	1	$N^{\text{te}}\|\mathcal{Y}\| + N^{\text{val}}$	$\geq 1-\alpha$
K-CV (4.12)	K	$N^{\text{te}}\|\mathcal{Y}\|K + N$	$\geq 1-2\alpha-\sqrt{2/N}$
CV (N-CV)	N	$N^{\text{te}}\|\mathcal{Y}\|N + N$	$\geq 1-2\alpha$
IFC (B.1)	$N^{\text{te}}\|\mathcal{Y}\|$	$N^{\text{te}}\|\mathcal{Y}\|(N+1)$	$\geq 1-\alpha$
EFC (B.2)	$N^{\text{te}}\|\mathcal{Y}\|(N+1)$	$N^{\text{te}}\|\mathcal{Y}\|(N+1)$	$\geq 1-\alpha$

Table 4.1 Computational Complexity and guarantees of Set predictions [90]

4.5 Online Conformal Prediction

In this section, we turn to online CP. Unlike the CP schemes presented in the previous section, online CP makes no assumptions about the probabilistic model underlying data generation [83, 84]. Rather, it models the observations as a deterministic stream of input-output pairs $z[i] = (x[i], y[i])$ over time index $i = 1, 2, \ldots$; and it targets a coverage condition defined in terms of the empirical rate at which the prediction set Γ_i at time i covers the correct output $y[i]$.

In the offline version of CP reviewed in the previous section, all N samples of the data set $\mathcal{D}$ are assumed to be available upfront (see Fig. 2.5(a)). In contrast, in online CP, a set predictor Γ_i for time index i is produced for each new input $x[i]$ over time $i = 1, 2, \ldots$ Specifically, given the past observations $\{z[j]\}_{j=1}^{i-1}$, the set predictor $\Gamma_i\Big(x[i]\Big|\{z[j]\}_{j=1}^{i-1}\Big)$ outputs a subset of the output space $\mathcal{Y}$. Given a target miscoverage level $\alpha \in [0, 1]$, an online set predictor is said to be $(1-\alpha)$-*long-term valid* if the following limit holds

$$\lim_{I\to\infty} \frac{1}{I}\sum_{i=1}^{I} \mathbb{1}\Big(y[i] \in \Gamma_i\Big(x[i]\Big|\{z[j]\}_{j=1}^{i-1}\Big)\Big) = 1-\alpha \tag{4.16}$$

for all possible sequences $z[i]$ with $i = 1, 2, \ldots$ Note that the condition (4.16), unlike (2.32), does not involve any ensemble averaging with respect to the data distribution. We will take (4.16) as the relevant definition of calibration for online learning.

Rolling conformal inference (RCI) [84] adapts in an online fashion a *calibration parameter* $\theta[i]$ across the time index i as a function of the instantaneous error variable

$$\text{err}[i] = \mathbb{1}\Big(y[i] \;\notin\; \Gamma_i(x[i])\Big), \tag{4.17}$$

which equals 1 if the correct output value is not included in the prediction set $\Gamma_i(x[i])$, and 0 otherwise. This is done using the update rule

$$\theta[i+1] \leftarrow \theta[i] + \gamma\Big(\text{err}[i] - \alpha\Big), \tag{4.18}$$

where $\gamma > 0$ is a learning rate. Accordingly, the parameter θ is increased by $\gamma(1-\alpha)$ if an error occurs at time i, and is decreased by $\gamma\alpha$ otherwise. Intuitively, a large positive parameter $\theta[i]$ indicates that the set predictor should be more inclusive in order to meet the validity constraint (4.16); and vice versa, a large negative value of $\theta[i]$ suggests that the set predictor can reduce the size of the prediction sets without affecting the long-term validity constraint (4.16).

Following [84], we elaborate on the use of the calibration parameter $\theta[i]$ in order to ensure condition (4.16) for an online version of the naïve quantile-based predictor (4.8) for scalar regression. A similar approach applies more broadly (see [83, 165], and [166]). Denote the data set $\mathcal{D}[i] = \{z[j]\}_{j=1}^{i-1}$ as having all previously observed labeled data set up till time $i-1$. The key idea behind RCI is to extend the naïve prediction interval (4.8) depending on the calibration parameter $\theta[i]$ as

$$\Gamma_i^{\text{RCI}}\Big(x[i]\Big|\mathcal{D}[i]\Big) = \Big[\hat{y}(x|\phi_{\mathcal{D}[i],\alpha/2}) - \varphi(\theta[i]), \hat{y}(x|\phi_{\mathcal{D}[i],1-\alpha/2}) + \varphi(\theta[i])\Big], \tag{4.19}$$

where

$$\varphi(\theta) = \text{sign}(\theta)\Big(\exp^{|\theta|} - 1\Big) \tag{4.20}$$

is the so-called stretching function, a fixed monotonically increasing mapping.

The set predictor RCI (4.19) "corrects" the NQB set predictor (4.8), via the additive stretching function $\varphi(\theta[i])$ based on the calibration parameter $\theta[i]$. As the time index i rolls, the calibration parameter $\theta[i]$ adaptively inflates and deflates according to (4.18). Upon each observation of new label $y[i]$, the quantile predictor model parameters $\phi_{\mathcal{D}[i],\alpha/2}$ and

$\phi_{\mathcal{D}[i],1-\alpha/2}$ can also be updated, without affecting the long-term validity condition (4.16) [84, Theorem 1]. We refer to Appendix B.2 for further details on online CP.

For the experimental part, we start with a toy regression problem, highlighting how misspecification affect set prediction. We then investigate demodulation of symbols using a moderate sized neural network, and later using deep learning modulation classification.

4.6 Toy Experiment for Regression

We investigate a regression example similar to [90]. Covariate x is in $\mathbb{R}^d$ and the target y is a real number. N such pairs $\mathcal{D} = \{(x[i], y[i])\}_{i=1}^N$, and N^{te} test evaluation pairs are drawn using the ground-truth Gaussian hierarchical model

$$p_0(\mathcal{D}, \phi) = p_0(\phi) \prod_{i=1}^{N} p_0(x[i]) p_0(y[i]|x[i], \phi) \tag{4.21}$$

where

$$\begin{aligned} p_0(\phi) &= \mathcal{N}(\phi|\mu_0 \mathbf{1}_d, \gamma_0^{-1} I_d), && \text{(4.22a)} \\ p_0(x) &= \mathcal{N}(x|0, d^{-1} I_d), && \text{(4.22b)} \\ p_0(y|x, \phi) &= \mathcal{N}(y|\phi^\top x, \beta_0^{-1}), && \text{(4.22c)} \end{aligned}$$

for ground-truth mean value $\mu_0 = 1$ and ground-truth precision values $\gamma_0 = 1$ and $\beta_0 = 1$, and $\mathbf{1}_d$ being the d-vector of all ones. This is repeated for $S = 100$ independent runs, in each a model parameter ϕ_s, N points data set $\mathcal{D}_s$, and N^{te} test pairs set $\left\{(x_s^{\text{te}}[j], y_s^{\text{te}}[j])\right\}_{j=1}^{N^{\text{te}}}$ are generated following (4.22). The set prediction coverage of any set prediction method is approximated by using the empirical average over the S independent runs

$$\mathbb{P}\Big(\mathbf{y} \in \Gamma_{\boldsymbol{\mathcal{D}},\alpha}(\mathbf{x})\Big) \approx \tfrac{1}{S} \sum_{s=1}^{S} \tfrac{1}{N^{\text{te}}} \sum_{j=1}^{N^{\text{te}}} \mathbb{1}\Big(y_s^{\text{te}}[j] \in \Gamma_{\mathcal{D}_s,\alpha}(x_s^{\text{te}}[j])\Big), \tag{4.23}$$

and the inefficiency, which is the set prediction average interval width, is similarly approximated as

$$\mathrm{E}_{(\boldsymbol{\mathcal{D}},\mathbf{x})\sim p(\mathcal{D},x)}\Big|\Gamma_{\boldsymbol{\mathcal{D}},\alpha}(\mathbf{x})\Big)\Big| \approx \tfrac{1}{S} \sum_{s=1}^{S} \tfrac{1}{N^{\text{te}}} \sum_{j=1}^{N^{\text{te}}} \Big|\Gamma_{\mathcal{D}_s,\alpha}(x_s^{\text{te}}[j])\Big)\Big|. \tag{4.24}$$

For prediction using frequentist learning, the ML predictor assumes the model

$$p(y|x, \phi) = \mathcal{N}(y|\phi^\top x, \beta^{-1}), \tag{4.25}$$

which in general can be misspecified when the precision $\beta \neq \beta_0$. The frequentist ML predictor under assumption (4.25) is given analytically for the overdetermined case ($N \geq d$) as the least squares solution

$$\phi_{\mathcal{D}}^{\mathrm{ML}} = (X_{\mathcal{D}}^{\top} X_{\mathcal{D}})^{-1} X_{\mathcal{D}}^{\top} y_{\mathcal{D}}, \tag{4.26}$$

where the data matrix $X_{\mathcal{D}} \in \mathbb{R}^{N \times d}$ stacks the N input vectors $\{x[i]\}_{i=1}^{N}$ on its rows, the label vector $y_{\mathcal{D}} \in \mathbb{R}^{N}$ stacks its targets. Further assuming a generally misspecified (when the mean $\mu \neq \mu_0$ or precision $\gamma \neq \gamma_0$) model parameter prior

$$p(\phi) = \mathcal{N}(\phi | \mu \mathbf{1}_d, \gamma^{-1} I_d), \tag{4.27}$$

and denoting the matrix

$$\Sigma_{\mathcal{D}}^{\mathrm{MAP}} = \Big(\gamma I_d + \beta X_{\mathcal{D}}^{\top} X_{\mathcal{D}}\Big)^{-1}, \tag{4.28}$$

allow for exact MAP solution

$$\phi_{\mathcal{D}}^{\mathrm{MAP}} = \Sigma_{\mathcal{D}}^{\mathrm{MAP}} \Big(\gamma \mu \mathbf{1}_d + \beta X_{\mathcal{D}}^{\top} y_{\mathcal{D}}\Big). \tag{4.29}$$

For the frequentist approach, any of the set predictors can use the NC scores with fitting algorithm of either ML (4.26) or MAP (4.29).

For predictions using Bayesian learning, the predictive posterior $p(y|x, \mathcal{D})$ can be obtained analytically for this toy example. The posterior, using the MAP vector (4.29) and MAP covariance matrix (4.28), is

$$p(\phi | \mathcal{D}) = \mathcal{N}(\phi | \phi_{\mathcal{D}}^{\mathrm{MAP}}, \Sigma_{\mathcal{D}}^{\mathrm{MAP}}). \tag{4.30}$$

The predictive posterior $p(y|x, \mathcal{D})$ follows

$$\begin{aligned} p(y|x, \mathcal{D}) &= \mathrm{E}_{\boldsymbol{\phi} \sim p(\phi|\mathcal{D})}[p(y|x, \boldsymbol{\phi})] \\ &= \mathcal{N}\Big(y \Big| (\phi_{\mathcal{D}}^{\mathrm{MAP}})^{\top} x, \beta^{-1} + x^{\top} \Sigma_{\mathcal{D}}^{\mathrm{MAP}} x\Big), \end{aligned} \tag{4.31}$$

whose log-loss leads to

$$\begin{aligned} \mathrm{NC}((x,y)|\mathcal{D}) &= -\log\Big(p(y|x, \mathcal{D})\Big) \\ &= -\log\Big(\mathcal{N}\Big(y \Big| (\phi_{\mathcal{D}}^{\mathrm{MAP}})^{\top} x, \beta^{-1} + x^{\top} \Sigma_{\mathcal{D}}^{\mathrm{MAP}} x\Big)\Big), \end{aligned} \tag{4.32}$$

which can be used as the Bayesian NC score for any of the set predictions.

We also include the set soft predictor, which is tractable for the hierarchical case. The frequentist ML and MAP assume fixed variance β^{-1} around the linear predictors $\phi_{\mathcal{D}}^{\text{ML}}$ and $\phi_{\mathcal{D}}^{\text{MAP}}$ respectively, while the soft Bayesian compensates estimation error of the MAP linear predictor by increasing the variance that now depends on the input X and data $\mathcal{D}$ to $\beta^{-1} + x^\top \Sigma_{\mathcal{D}}^{\text{MAP}} x$. Since all distributions are Gaussian, the set lies in an interval with width inversely proportional to the standard deviation. We have the set soft ML predictor

$$\Gamma_{\mathcal{D},\alpha}^{\text{soft-ML}}(x) = \left[(\phi_{\mathcal{D}}^{\text{ML}})^\top x - \tfrac{1}{2} w^{\text{soft-ML}}, (\phi_{\mathcal{D}}^{\text{ML}})^\top x + \tfrac{1}{2} w^{\text{soft-ML}}\right] \tag{4.33}$$

with set width

$$w^{\text{soft-ML}} = 2\sqrt{2\beta^{-1}}\text{erf}^{-1}(1-\alpha) \tag{4.34}$$

and $\text{erf}^{-1}(\cdot)$ as the inverse to the error function

$$\text{erf}(y) = \tfrac{2}{\sqrt{\pi}} \int_0^y e^{-t^2} \text{d}t. \tag{4.35}$$

The soft-MAP is similar to (4.34), using MAP predictor in lieu of ML, and has the same width $w^{\text{soft-MAP}} = w^{\text{soft-ML}}$. The soft-Bayes set predictor follows

$$\Gamma_{\mathcal{D},\alpha}^{\text{Bay}}(x) = \left[(\phi_{\mathcal{D}}^{\text{MAP}})^\top x - \tfrac{1}{2} w^{\text{soft-MAP}}(x), (\phi_{\mathcal{D}}^{\text{MAP}})^\top x + \tfrac{1}{2} w^{\text{soft-MAP}}(x)\right] \tag{4.36}$$

and has a larger width

$$w^{\text{soft-MAP}}(x) = 2\sqrt{2(\beta^{-1} + x^\top \Sigma_{\mathcal{D}}^{\text{MAP}} x)}\text{erf}^{-1}\Big(1-\alpha\Big), \tag{4.37}$$

hence being more inefficient yet has better coverage.

Fig. 4.5 and Fig. 4.6 show how misspecification affects coverage and inefficiency. Specifically, we use assumed model prior mean parameter $\mu = 1$ and assumed likelihood precision $\beta = 1$, matched to the ground truth values $\mu_0 = 1$ and $\beta_0 = 1$. We sweep the assumed model prior precision γ around its ground truth value $\gamma_0 = 1$. In the proximity of the well-specified case (the central point), the Bayes set prediction in each conformal method is more efficient than its frequentist-based set predictors. While the former depends on the assumed parameter γ, the latter does not, which may result in ML solution more efficient than the Bayesian when the prior is poorly set.

The soft set predictors produce more efficient sets, while its ML set achieves the genie-aided inefficiency, governed by the residual noise of (4.22c). This however, is insignificant, since the soft set predictor does not hold validity, excluding when well-specified which can

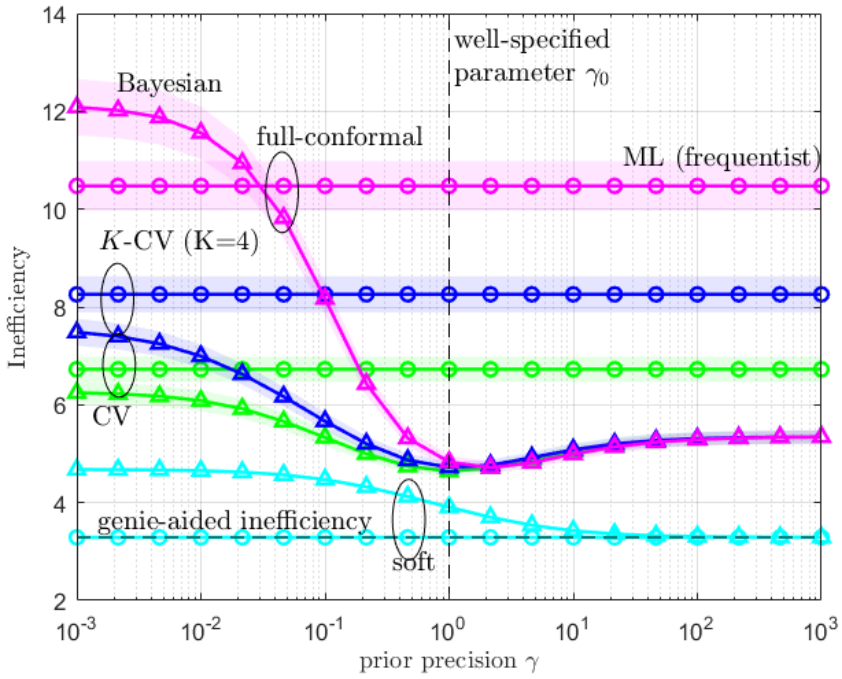


Fig. 4.5 Set prediction interval width as function of misspecified for the toy regression problem (Section 4.6). Total of $S = 100$ test trials and $N^{\text{te}} = 100$ test points in each. Confidence intervals are $\pm$ one standard deviation. For each set predictor, the frequentist NC scores using ML are marked with circles, while the Bayesian NC score using the exact Hessian are in triangles.

be achieved in toy examples but rarely in real life problems. Validity for all other methods retains even for great misspecification levels, through the adaptation of wider prediction sets.

4.7 Experiment of Symbol Demodulation

In this section, we focus on the application of offline CP, as described in Sec. 4.4, to the problem of symbol demodulation in the presence of transmitter hardware imperfections. This problem was also considered in [49, 25] by focusing on frequentist and Bayesian learning. Unlike [49, 25], we investigate the use of CP as a means to obtain set predictors satisfying the validity condition (2.32).

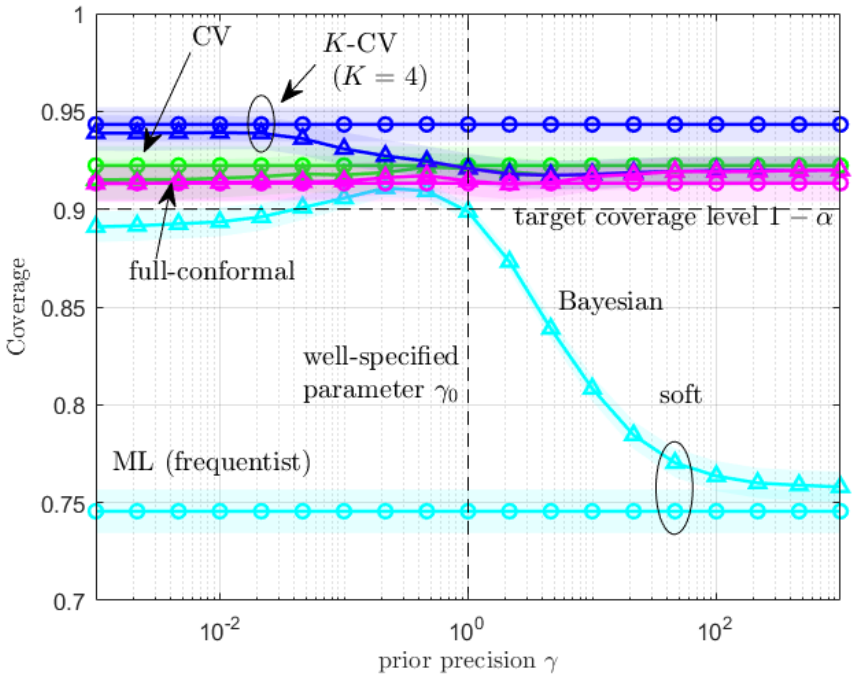


Fig. 4.6 Set prediction coverage corresponding to Fig 4.5 setting.

4.7.1 Problem Formulation

The problem of interest consists of the demodulation of symbols from a discrete constellation based on received baseband signals subject to hardware imperfections, noise, and fading. The goal is to design *set demodulators* that output a subset of all possible constellation points with the guarantee that the subset includes the true transmitted signal with the desired target probability $1-\alpha$. Set-valued demodulation can be useful when combined with channel decoders that can concurrently explore several paths [167]. Furthermore, while we do not investigate this direction here, CP could be applied directly to decoding, yielding a form of list decoding (see, e.g., [168]) with formal reliability guarantees.

To keep the notation consistent with the previous sections, we write as $y[i]$ the i-th transmitted symbols, and as $x[i]$ the corresponding received signal. Each transmitted symbol $y[i]$ is drawn uniformly at random from a given constellation $\mathcal{Y}$. We model I/Q imbalance at the transmitter and phase fading as in [112]. Accordingly, the ground-truth channel law connecting symbols $y[i]$ into received samples $x[i]$ is described by the equality

$$\mathbf{x}[i] = e^{\boldsymbol{\mathcal{W}}} f_{\mathrm{IQ}}(\mathbf{y}[i]) + \mathbf{v}[i], \tag{4.38}$$

for a random phase $\boldsymbol{\psi} \sim \mathrm{U}[0, 2\pi)$, where the additive noise is $\mathbf{v}[i] \sim \mathcal{CN}(0, \mathrm{SNR}^{-1})$ for signal-to-noise ratio level SNR. Furthermore, the I/Q imbalance function [137] is defined as

$$f_{\mathrm{IQ}}(\mathbf{y}[i]) = \bar{\mathbf{y}}_{\mathrm{I}}[i] + \jmath \bar{\mathbf{y}}_{\mathrm{Q}}[i], \tag{4.39}$$

where

$$\begin{bmatrix} \bar{\mathbf{y}}_{\mathrm{I}}[i] \\ \bar{\mathbf{y}}_{\mathrm{Q}}[i] \end{bmatrix} = \begin{bmatrix} 1+\boldsymbol{\epsilon} & 0 \\ 0 & 1-\boldsymbol{\epsilon} \end{bmatrix} \begin{bmatrix} \cos\boldsymbol{\delta} & -\sin\boldsymbol{\delta} \\ -\sin\boldsymbol{\delta} & \cos\boldsymbol{\delta} \end{bmatrix} \begin{bmatrix} \mathbf{y}_{\mathrm{I}}[i] \\ \mathbf{y}_{\mathrm{Q}}[i] \end{bmatrix}, \tag{4.40}$$

with $\mathbf{y}_{\mathrm{I}}[i]$ and $\mathbf{y}_{\mathrm{Q}}[i]$ being the real and imaginary parts of the modulated symbol $\mathbf{y}[i]$; and $\bar{\mathbf{y}}_{\mathrm{I}}[i]$ and $\bar{\mathbf{y}}_{\mathrm{Q}}[i]$ standing for the real and imaginary parts of the transmitted symbol $f_{\mathrm{IQ}}(\mathbf{y}[i])$. In (4.40), the channel state $\mathbf{c}$ consists of the tuple $\mathbf{c} = (\boldsymbol{\psi}, \boldsymbol{\epsilon}, \boldsymbol{\delta})$ encompassing the complex phase $\boldsymbol{\psi}$ and the I/Q imbalance parameters $(\boldsymbol{\epsilon}, \boldsymbol{\delta})$.

4.7.2 Implementation

As in [49, 25], demodulation is implemented via a neural network probabilistic model $p(y|x, \phi)$ consisting of a fully connected network with real inputs $x[i]$ of dimension 2 as per (4.38), followed by three hidden layers with $10, 30$, and 30 neurons having ReLU activations in each layer. The last layer implements a softmax classification for the $|\mathcal{Y}|$ possible constellation points.

We adopt the standard NC score (4.9), where the trained model $\phi_{\mathcal{D}}$ for frequentist learning is obtained via $I = 120$ GD update steps for the minimization of the cross-entropy training loss with learning rate $\eta = 0.2$; while for Bayesian learning we implement a gradient-based MC method, namely Langevin MC, with burn-in period of $R_{\min} = 100$, ensemble size $R = 20$, learning rate $\eta = 0.2$, and temperature parameter $T = 20$. We assume standard Gaussian distribution for the prior distribution [39]. Details on Langevin MC can be found in Appendix B.3.

In the experiments, we adopted Langevin MC to approximate the Bayesian posterior [39, 28]. Langevin MC adds Gaussian noise to each standard GD update for frequentist learning (see, e.g., [28, Sec. 4.10]). The noise has power $2\eta/T$, where η is the GD learning rate and $T > 0$ is a temperature parameter. Langevin MC produces R model parameters $\{\phi[r]\}_{r=1}^{R}$ across R consecutive iterations. We specifically retain only the last R samples, discarding an initial burn-in period of $R_{\min}$ iterations. The temperature parameter T is typically chosen to be larger than 1 [169, 170]. With the R samples, the expectation term in (2.20) is approximated as the empirical average $\frac{1}{R}\sum_{r=1}^{R} p(y|x, \phi[r])$.

We observe that Langevin MC is a probabilistic training algorithm, and that it satisfies the permutation-invariance property in terms of the distribution of the random output models discussed in Sec. 4.4.3.

We compare the naïve set predictor (4.3), also studied in [49, 25], which provides no formal coverage guarantees, with the CP set prediction methods reviewed in Sec. 4.4. VB-CP uses equal set sizes for the training and validation sets. We target the miscoverage level as $\alpha = 0.1$.

4.7.3 Results

We consider the Amplitude and Phase-Shift Keying (APSK) modulation with $|\mathcal{Y}| = 8$. The SNR level is set to SNR $= 5$ dB. The amplitude and phase imbalance parameters are independent and distributed as $\boldsymbol{\epsilon} \sim \text{Beta}(\epsilon/0.15|5, 2)$ and $\boldsymbol{\delta} \sim \text{Beta}(\delta/15^\circ|5, 2)$, respectively [49].

Fig. 4.7 shows the

$$\text{empirical coverage} = \frac{1}{N^{\text{te}}} \sum_{j=1}^{N^{\text{te}}} \mathbb{1}\Big(y^{\text{te}}[j] \in \Gamma(x^{\text{te}}[j]|\mathcal{D})\Big), \tag{4.41}$$

and Fig. 4.8 shows the

$$\text{empirical inefficiency} = \frac{1}{N^{\text{te}}} \sum_{j=1}^{N^{\text{te}}} \Big|\Gamma(x^{\text{te}}[j]|\mathcal{D})\Big|, \tag{4.42}$$

both evaluated on a test set $\mathcal{D}^{\text{te}} = \{(x^{\text{te}}[j], y^{\text{te}}[j])\}_{j=1}^{N^{\text{te}}}$ with $N^{\text{te}} = 100$ data points, as a function of the size of the available data set $\mathcal{D}$. We average the results for 50 independent trials, each corresponding to independent draws of the variables $\{\mathcal{D}, \mathcal{D}^{\text{te}}\}$ from the ground truth distribution. This way, the metrics (4.41)-(4.42) provide an estimate of the coverage (2.32) and of the inefficiency (2.33), respectively [90].

From Fig. 4.7, we first observe that the naïve set predictor, with both frequentist and Bayesian learning, does not meet the desired coverage level in the regime of a small number N of available samples. In contrast, confirming the theoretical calibration guarantees presented in Sec. 4.4, all CP methods provide coverage guarantees, achieving coverage rates above $1 - \alpha$. Furthermore, as seen in Fig. 4.8, coverage guarantees are achieved by suitably increasing the size of prediction sets, which is reflected by the larger inefficiency. The size of the prediction sets, and hence the inefficiency, decreases as the data set size, N, increases. In this regard, due to their more efficient use of the available data, CV-CP and K-CV-CP

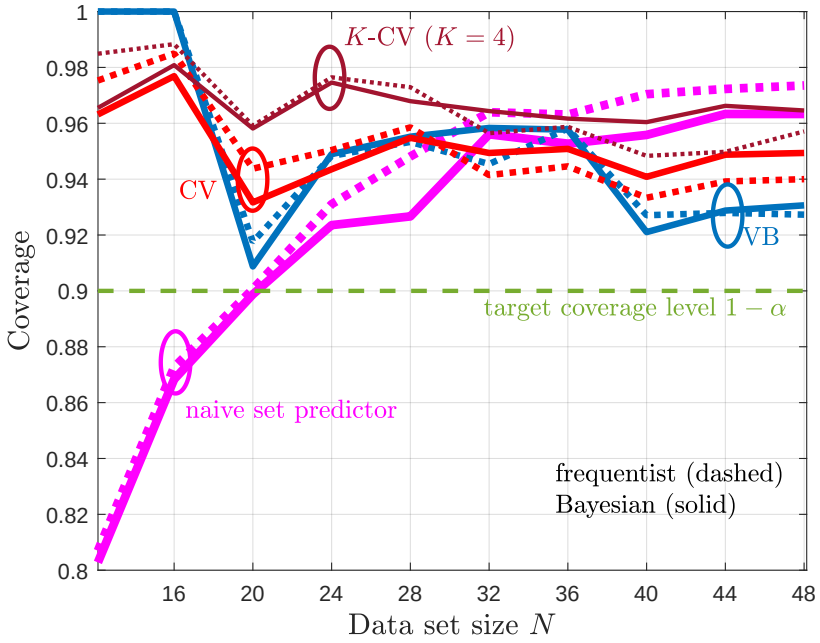


Fig. 4.7 Coverage for naïve set predictor (4.3), VB-CP (4.10), CV-CP, and K-CV-CP (4.12) with $K = 4$, for symbol demodulation problem (Section 4.7). For every set predictors, the NC scores are evaluated either using frequentist learning (dashed lines) or Bayesian learning (solid lines). The coverage level is set to $1 - \alpha = 0.9$, and each numerical evaluation is averaged over 50 independent trials (new channel state c) with $N^{\text{te}} = 100$ test points.

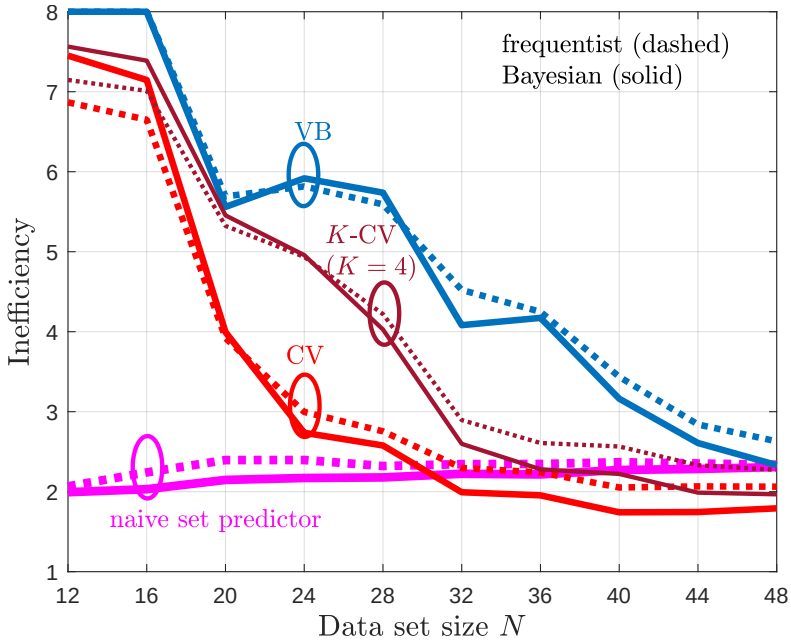


Fig. 4.8 Average set prediction size (inefficiency) for the same setting of Fig. 4.7.

predictors have a lower inefficiency as compared to VB predictors, with CV-CP offering the best performance. Finally, Bayesian NC scores are generally seen to yield set predictors with lower inefficiency, confirming the merits of Bayesian learning in terms of calibration.

4.8 Experiment of Modulation Classification

In this section, we propose and evaluate the application of offline CP to the problem of modulation classification [68, 69].

4.8.1 Problem Formulation

Due to the scarcity of frequency bands, electromagnetic spectrum sharing among licensed and unlicensed users is of special interest to improve the efficiency of spectrum utilization. In sensing-based spectrum sharing, a transmitter scans the prospective frequency bands to identify, for each band, if the spectrum is occupied, and, if so, if the signal is from a licensed user or not. A key enabler for this operation is the ability to classify the modulation of the received signal [171]. The modulation classification task is made challenging by the dimensionality of the baseband input signal and by the distortions caused by the propagation channel. Data-driven solutions [172] have shown to be effective for this problem in terms of accuracy, while the focus here is on calibration performance.

Accordingly, we aim at designing *set modulation classifiers* that output a subset of the set of all possible modulation schemes with the property that the true modulation scheme is contained in the subset with a desired probability level $1 - \alpha$. Set predictions provide actionable information in several applications of modulation classifiers. For instance, in cognitive radio systems, there may be protected modulation schemes adopted by a primary user [173, 174]. If the predicted set produced by a secondary user includes such schemes, the secondary user should, e.g., refrain from transmitting or use lower transmission powers, guaranteeing that the primary user is not affected with probability at least $1 - \alpha$. As another example, in military or emergency response applications, modulation classification is used a preliminary step to synchronize a receiver [175, 176]. In this case, having a list of possible modulation schemes enables the receiver to attempt decoding in parallel on all possible identified schemes, with the guarantee that the correct modulation scheme is evaluated with probability at least $1 - \alpha$.

We adopt the data set provided by [69], which has approximately 2.5×10^6 baseband signals, each produced using one out of 24 possible digital and analog modulations across different SNR values and channel models. The data set contains data for single-carrier

modulations such as OOK, ASK, PSK, APSK, QAM, AM, FM, GMSK, and OQPSK, including several modulation orders for each scheme. For each modulation, several short-time signal observations of 1024 I/Q samples are synthesized using random realizations of Rayleigh fading, carrier frequency mismatch, sampling rate mismatch, and shaping roll-off values. Out of the whole data set, here we focus on the high SNR regime (≥ 6 dB). Accordingly, the data set $\mathcal{D}$ consists of approximately 1.28×10^6 pairs (x, y), where x is the channel output signal of 2048 interleaved I/Q samples and y is the index of one of the $|\mathcal{Y}| = 24$ possible modulations. The SNR value itself is not available to the classifier.

4.8.2 Implementation

We use a neural network architecture similar to the one used in [69], which has 7 one-dimensional convolutional layers with kernel size 3 and 64 channels for all layers, except for the first layer which has 2 channels. The convolution layers are followed by 3 fully-connected linear layers. A scaled exponential linear unit (SELU) is used for all inner layers, and a softmax is used at the last, fully connected, layer. We assume availability of $N = 4800$ pairs (x, y) for the data set $\mathcal{D}$, while gauging the empirical inefficiency and coverage level with $N^{\text{te}} = 1000$ held-out pairs. A total number of $I = 4000$ GD steps with fixed learning rate of 0.02 are carried out, and the target miscoverage rate is set to $\alpha = 0.1$. VB partitions its available data into equal sets for training and validation.

4.8.3 Results

In this problem, due to computational cost, we exclude CV-CP and we focus on K-CV-CP with a moderate number of folds, namely $K = 6$ and $K = 12$. In Fig. 4.9, box plots show the quartiles of the empirical coverage (4.41) and of the empirical inefficiency (4.42) from 32 independent runs, with different realizations of data set and test examples. The lower edge of the box represents the 0.25-quantile; the solid line within the box the median; the dashed line within the box the average; and the upper edge of the box the 0.75-quantile. As can be seen in the figure, the naïve set predictor is invalid (see average shown as dashed line), and it exhibits a wide spread of the coverage rates across the trials. On the other hand, all CP set predictors are valid, meeting the predetermined coverage level $1 - \alpha = 0.9$, and have less spread-out coverage rates.

As also noted in the previous section, VB-CP suffers from larger predicted set size as compared to K-CV-CP, due to poor sample efficiency. A small number of folds, as low as $K = 6$, is sufficient for K-CV-CP to outperform VB-CP. This improvement in efficiency

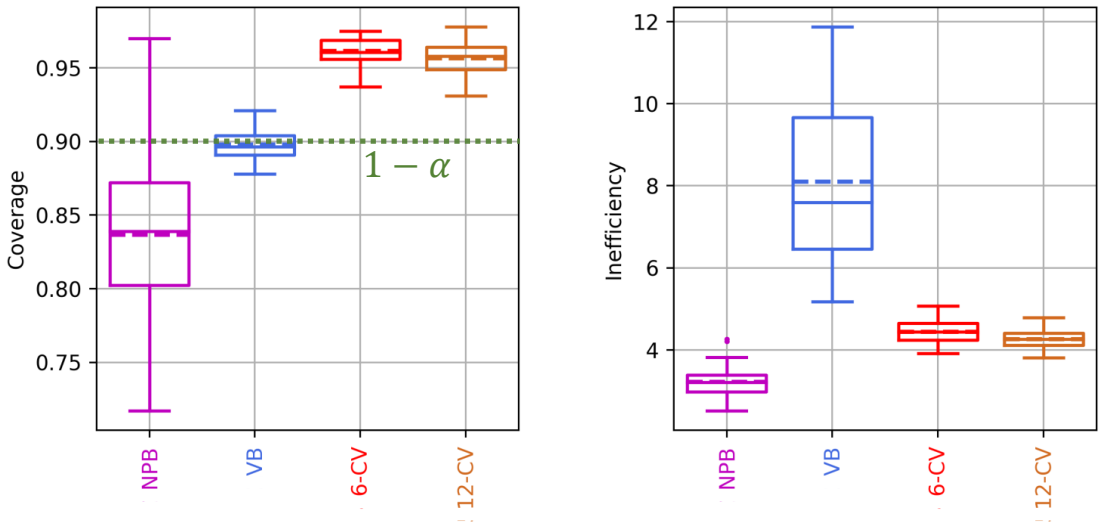


Fig. 4.9 Coverage and inefficiency for NPB (4.3), VB-CP (4.10), and K-CV-CP (4.12) with $K = 6$ and $K = 12$, for the modulation classification problem (implementation details in Section 4.8.2). The boxes represent the 25% (lower edge), 50% (solid line within the box), and 75% (upper edge) percentiles of the empirical performance metrics evaluated over 32 different experiments, with average value shown by the dashed line.

comes at the computational cost of training six models, as compared to the single model trained by VB-CP.

4.9 Experiment of RSS using Online Channel Prediction

In this section, we investigate the use of online CP, as described in Sec. 4.5, for the problem of channel prediction. We specifically focus on the prediction of the received signal strength (RSS), which is a key primitive at the physical layer, supporting important functionalities such as resource allocation [177, 178].

4.9.1 Problem Formulation

Consider a receiver that has access to a sequence of RSS samples from a given device. We aim at designing a predictor that, given a sequence of past samples from the RSS sequence, produces an *interval* of values for the next RSS sample. To meet calibration requirements, the interval must contain the correct future RSS value with the desired rate level $1 - \alpha$.

Unlike the previous applications, here the rate of coverage is evaluated based on the time average

$$\frac{1}{t}\sum_{i=1}^{t} \mathbb{1}\Big(y[i] \in \Gamma_i\Big(x[i]\Big|\{z[j]\}_{j=1}^{i-1}\Big)\Big). \tag{4.43}$$

This is computed as the fraction previous time instants $i \in \{1, ..., t\}$ at which the set predictor Γ_i includes the true RSS value $y[i]$.

Interval predictions for communication channels can support resource allocation. For instance, a scheduler may select users whose lowest possible RSS in the predicted set is sufficiently large to enable communication at some desired level [179]. A scheduler may also use larger values in the predicted set to facilitate exploration via optimism [180, 181]. Finally, set prediction can enable the detection of outage events, or anomalies, by checking if the lowest allowed RSS is not in line with the past observations for a given link.

We consider two data sets of RSS sequences. The first data set records RSS samples $y[i]$ in logarithmic scale for an IEEE 802.15.4 radio over time index i [1]. We further use the available side information on the time-variant channel ID, which determines the carrier frequency used at time i out of the 16 possible bands, as the input $x[i]$. At time i, we observe a sequence of RSS samples $z[1], \ldots, z[i-1]$ with $z[i] = (x[i], y[i])$, and the goal is to predict the next RSS sample $y[i]$ via the online set predictor Γ_i^{RCI} (4.19).

The second data set [2] reports samples $y[i]$, measured in dBm, on a 5.8 GHz device-to-device link without additional input. Hence, in this case, we predict the next RSS sample $y[i]$ using the previous RSS samples $y[1], \ldots, y[i-1]$. Note that the prior works [1, 2] adopted standard probabilistic predictors, while here we focus on set predictors that produce a prediction interval $\Gamma_i^{\text{RCI}}\Big(x[i]\Big|\{z[j]\}_{j=1}^{i-1}\Big)$.

4.9.2 Implementation

We build the CP set predictor by leveraging the probabilistic neural network used in [84] as the model class for the quantile predictors in (4.7)-(4.8). Each quantile predictor consists of a multi-layer neural network that pre-processes the most recent K pairs $\{z[i-K], \ldots, z[i-1]\}$; of a stacked long short-term memory (LSTM) [182] with two layers; and of a post-processing neural network, which maps the last LSTM hidden vector into a scalar that estimates the quantile used in (4.8). For details of the implementation, we refer to Appendix B.4.

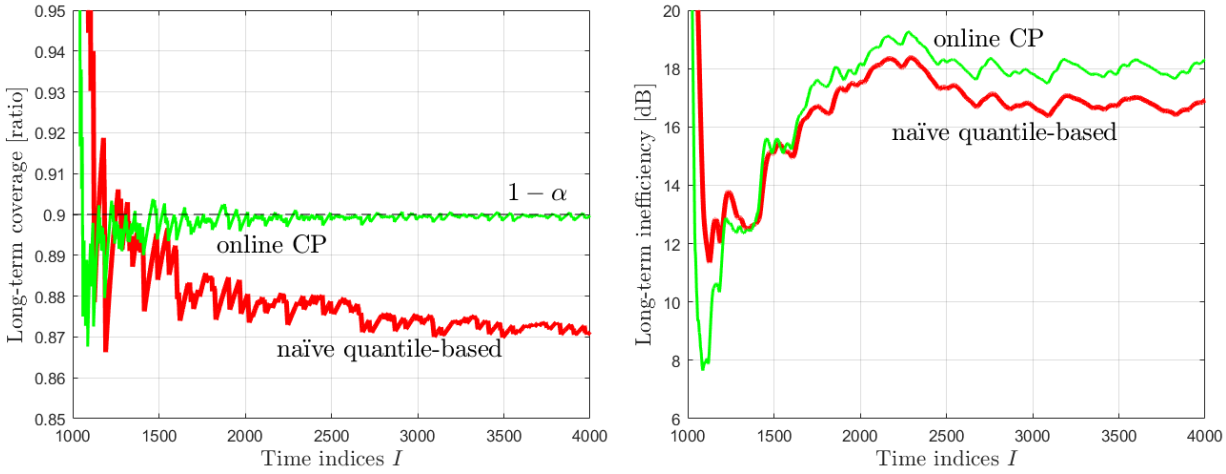


Fig. 4.10 CP for time-series `Outdoor` of [1]: **(left)** Time-average coverage (4.44) of naïve set prediction and online CP; **(right)** Time-averaged inefficiency (4.45) of naïve set prediction and online CP.

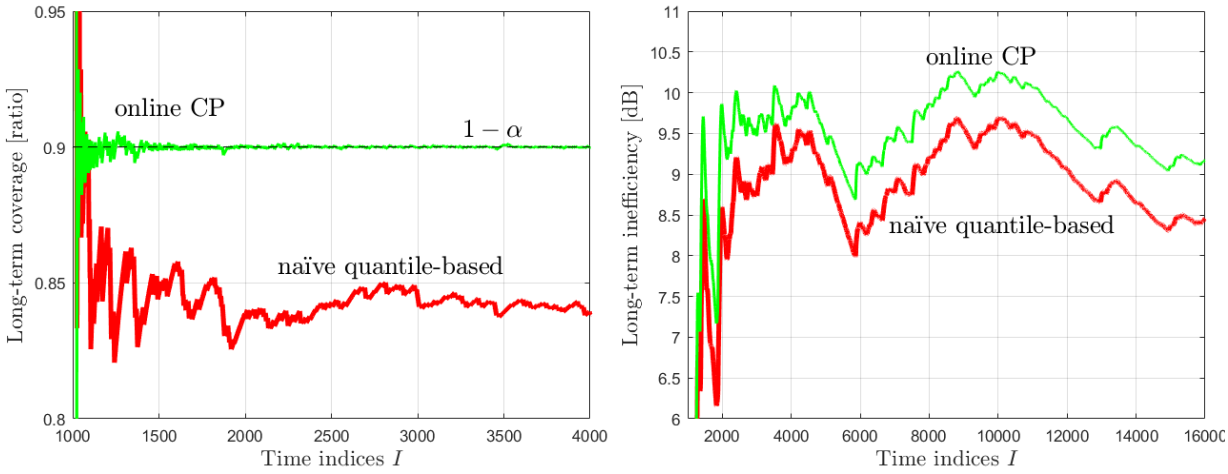


Fig. 4.11 CP for time-series `NLOS_Head_Indoor_1khz` in [2]: **(left)** Time-average coverage (4.44) of naïve set prediction and online CP; **(right)** Time-average inefficiency (4.45) of naïve set prediction and online CP.

4.9.3 Results

Fig. 4.10 and Fig. 4.11 report the

$$\text{time-average coverage} = \frac{1}{I}\sum_{i=1}^{I} \mathbb{1}\Big(y[i] \in \Gamma_i\Big(x[i]\Big|\{z[j]\}_{j=1}^{i-1}\Big)\Big) \tag{4.44}$$

and the

$$\text{time-average inefficiency} = \frac{1}{I}\sum_{i=1}^{I} \Big|\Gamma_i\Big(x[i]\Big|\{z[j]\}_{j=1}^{i-1}\Big)\Big| \tag{4.45}$$

for online CP (4.19), compared to a baseline of the naïve quantile-based predictor (4.8), as a function of the time window size I for data sets [2] and [2], respectively. We have discarded 1000 samples for a warm-up period for both metrics (4.44) and (4.45).

In both cases, the naïve predictor is seen to fail to satisfy the coverage condition (4.16) for both data sets, while online CP converges to the target level $1 - \alpha = 0.9$. This result is obtained by online CP with a modest increase of around 8% for both data sets in terms of inefficiency.

4.10 Conclusion

AI in communication engineering should not only target accuracy, but also calibration, ensuring a reliable and safe adoption of machine learning within the overall telecommunication ecosystem. In this chapter, we have proposed the adoption of a general framework, known as conformal prediction (CP), to *transform* any existing AI model into a well-calibrated model via post-hoc calibration for communication engineering. Depending on the situation of interest, post-hoc calibration leverages either an held-out (cross) validation set or previous samples. Unlike calibration approaches that do not formally guarantee reliability, such as Bayesian learning or temperature scaling, CP provides formal guarantees of calibration, defined either in terms of ensemble averages or long-term time averages. Calibration is retained irrespective of the accuracy of the trained models, with more accurate models producing smaller set predictions.

To validate the reliability of CP-based set predictors, we have provided extensive comparisons with conventional methods based on Bayesian or frequentist learning. Focusing on demodulation, modulation classification, and channel prediction, we have demonstrated that AI models calibrated by CP provide formal guarantees of reliability, which are practically essential to ensure calibration in the regime of limited data availability.

Chapter 5

Guaranteed Dynamic URLLC Scheduling via Conformal Prediction

5.1 Introduction

5.1.1 Motivation and overview

Servicing *ultra-reliable and low-latency communication* (URLLC) traffic typically calls for a pre-emptive allocation of resources in order to meet stringent delay constraints [183–185]. A conservative static allocation of resources for URLLC may guarantee desired levels of reliability and latency, but this comes at the expense of other services, most notably *enhanced mobile broadband (eMBB)*, which cannot use the resources reserved for URLLC. A dynamic allocation of resources, while potentially more efficient, is made challenging by the stochastic nature of URLLC data packet generation, particularly for the uplink [186, 184, 187, 188]. A promising solution is the adoption of predictors of URLLC data packet generation. Concretely, with reference to Fig. 5.1, a base station can deploy a predictor of URLLC data packet generation for the following frame, so as to guide the adaptive allocation of slots for URLLC packets, leaving the other slots available for eMBB users.

Such predictors may be based on models that leverage domain knowledge [189] or statistical information extracted from data [190]. In either case, predictions are bound to be imperfect due to model misspecification or to an insufficient access to data [190]. Therefore, predictors may consistently overestimate or underestimate the amount of URLLC data to be generated. As a consequence, schedulers that operate on the basis of such predictors

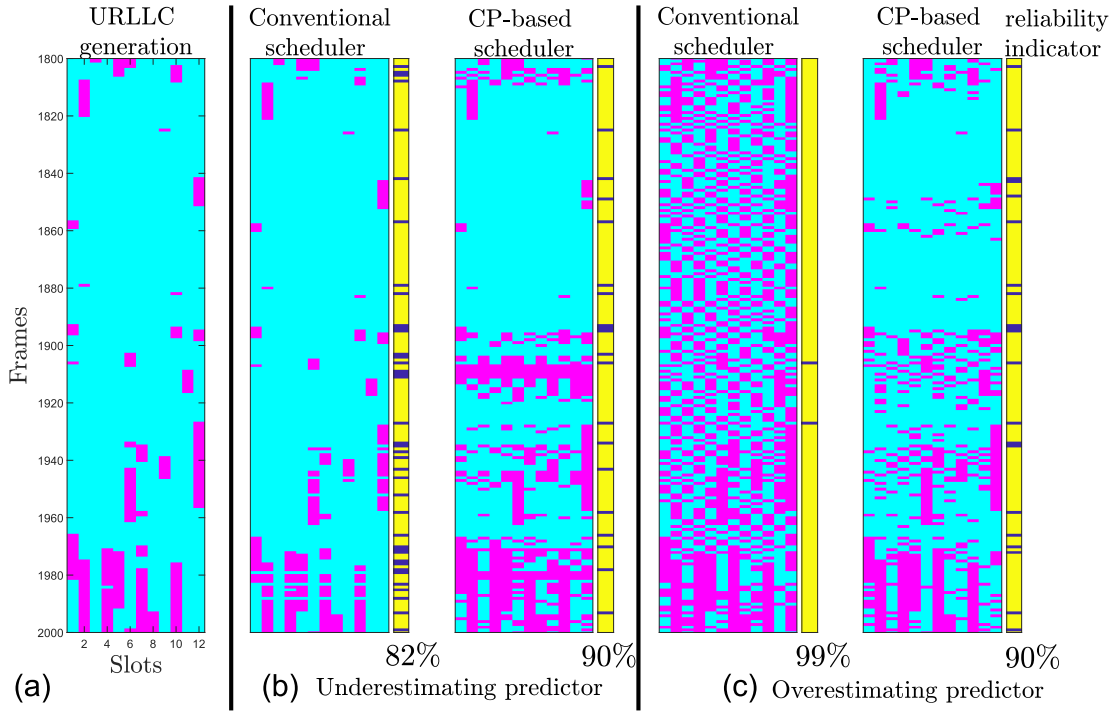


Fig. 5.1 **(a)** Data packet generation for URLLC traffic across successive frames (URLLC packets are shown in the darker color). This information is unavailable at the scheduler, which has access only to a predictor that may underestimate or overestimate the number of URLLC packets to be generated (as in parts **(b)** and **(c)** respectively). **(b)** In the former case, a conventional resource allocation scheme that trusts the predictor fails to reliably serve URLLC data (slots allocated for URLLC are in darker color), resulting in an average frame success ratio of 82% that falls short of the target of 90% (for illustrative purposes we set the target unreliability rate to be modest using $\alpha = 0.1$, our numerical part uses a tighter value). Scheduling error are shown as darker slots in the sidebar. **(c)** With an overestimating predictor, a conventional scheduler allocates excessive resources to URLLC traffic, severely impairing eMBB efficiency. eMBB traffic can occupy all slots unassigned to URLLC packets. In either case, the proposed CP-based scheduler is able to meet the URLLC reliability target of 90% by properly adjusting the eMBB spectral efficiency.

would yield either an excessive or an insufficient amount of resources to be pre-emptively allocated for URLLC packets in future frames (see Fig. 5.1 for an illustration).

In this chapter, we introduce a novel scheduler for URLLC packets that provides formal guarantees on reliability and latency *irrespective of the quality of the URLLC traffic predictor*. The proposed method leverages recent advances in *online conformal prediction (CP)* [83, 84], by dynamically adjusting the amount of allocated resources so as to meet reliability and latency requirements.

5.1.2 Related work

Model-based URLLC traffic predictors, which assume perfect knowledge on the traffic model for optimal allocation strategies, are studied in [191, 189, 186, 192–194, 149]. Data-driven approaches [190, 195–199], which observe data for model training for resource allocation, use tools including unsupervised learning [197], and online learning [198, 199].

CP is a class of post-hoc calibration methods that transform standard probabilistic model into a *set predictor* that is guaranteed to contain the true target with probability no smaller than a predetermined coverage level [200, 201]. CP is experiencing a renaissance [162, 90, 202, 117], with novel applications in [203–206]. *Online CP* alleviates the limitation of conventional CP of requiring a separate calibration data at the cost of providing time-averaged, rather than ensemble, reliability guarantees [83, 84, 165, 207]. The adoption of CP in communication engineering was proposed in [111], which focused on wireless applications such as symbol demodulation, modulation classification, and received signal strength prediction.

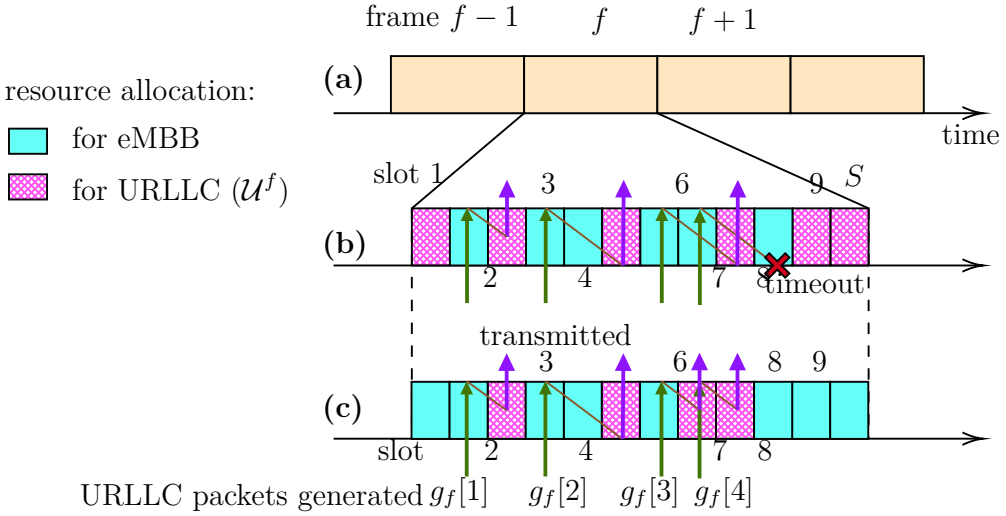


Fig. 5.2 **(a)** The assumed frame-based communication: each frame f contains S slots that can be allocated for either eMBB or URLLC traffic. **(b)** Illustration of the generation of $G_f = 4$ URLLC packets, with each packet generated at a slot marked by an upward arrow incoming into the frame. Each URLLC packet must find an available slot within a maximum delay of $L = 2$ slots in order to meet latency requirements. With the given resource allocation, the first three packets are transmitted in the corresponding slots indicated with an upward outgoing arrow, while the fourth packet does not find any available slot within the delay constraint. **(c)** For the illustrated distinct slot allocation, all URLLC packets are transmitted within the allowed latency of $L = 2$ slots.

5.2 System Model and Problem Definition

Fig. 5.2 illustrates the assumed frame-based transmission setting. Each frame consists of a set $\mathcal{S} = \{1, \ldots, S\}$ of S slots, and each of the slots can be allocated either to URLLC or eMBB packets. At the beginning of each frame f, a *scheduler* at the base station allocates a subset $\mathcal{U}_f \subseteq \mathcal{S}$ of slots for URLLC transmission, and remaining slots are devoted to eMBB traffic. The main challenge is that the scheduler does not know in advance when URLLC devices will generate packets [186, 184].

5.2.1 URLLC data generation

For any frame $f = 1, 2, \ldots$, a total of $G_f \leq S$ URLLC packets are generated. The i-th generated packet is produced in the $g_f[i] \in \mathcal{S}$ slot of the frame. As in [208] we make the simplifying assumption that no more than one URLLC packet can be generated in a slot. This assumption encodes the requirement that URLLC traffic can be successfully served within any desired degree of reliability by an ideal scheduler that knows the URLLC traffic pattern (or by a trivial scheduler that allocates all slots to URLLC transmissions). The slot indices at which URLLC packets are generated are collected in set $\mathcal{G}_f = \{g_f[1], \ldots, g_f[G_f]\}] \subseteq \mathcal{S}$. Importantly, no further assumptions are made on the URLLC data generation mechanism.

5.2.2 URLLC latency and reliability constraints

The goal of the scheduler is to allocate the smallest number $U_f = |\mathcal{U}_f|$ of slots, while ensuring that URLLC traffic is served with a prescribed level of latency and reliability. Note that the proposed approach is in line with 3GPP's preemptive scheduling of URLLC traffic on top of eMBB transmissions [209]. Specifically, *latency constraints* impose that an URLLC packet generated in time slot $s \in \mathcal{S}$ must be allocated a time slot in the interval $[s, s+1, \ldots, \min\{s+L, S\}]$ given maximum allowed latency of L slots. *Reliability* is measured by the fraction of frames f in which *all* G_f URLLC packets are allocated a slot within the described latency constraint of L slots. In particular, we impose that the fraction of frames satisfying this condition is at least $1-\alpha$, for some *unreliability rate* $\alpha \in (0, 1)$.

To formalize the outlined latency and reliability constraints, we introduce the following definition. We say that a subset $\mathcal{U}_f$ of allocated slots in frame f *"L-covers"* a subset $\mathcal{G}_f$ of slots at which URLLC packets are generated if the following condition is met: For each generated URLLC packet $g \in \mathcal{G}_f$, there is a *distinct* allocated URLLC slot $u \in \mathcal{U}_f$ within the latency constraint L, i.e., such that the inequalities $0 \leq u - g \leq L$ are satisfied. Note

that this condition implies that the number of allocated slots is no smaller than the number of generated packets, i.e., $|\mathcal{U}_f| \geq |\mathcal{G}_f|$.

As an example, in Fig. 5.2(b) the allocation $\mathcal{U}_f = \{1, 3, 6, 9, 11, 12\}$ fails to "2-cover" the generated set $\mathcal{G}_f = \{2, 4, 7, 8\}$ since the packet generated at $g_f[4] = 8$ cannot be served within the latency constraint $L = 2$. The allocated slot 9 "covers" the packet generated at $g_f[3] = 7$ and hence is unavailable for $g_f[4] = 8$, while the remaining allocated slots 11 and 12 do not meet the latency constraint. In contrast, the URLLC allocation in Fig. 5.2(c) succeeds in 2-covering the same generated packet $\mathcal{G}_f$.

Given the set of generated packets $\mathcal{G}_f$ and the set of URLLC allocated sets $\mathcal{U}_f$, the reliability measure for frame f is set as the indicator

$$r(\mathcal{U}_f|\mathcal{G}_f) = \begin{cases} 1 & \text{if } \mathcal{U}_f \ L\text{-covers } \mathcal{G}_f \\ 0 & \text{otherwise.} \end{cases} \tag{5.1}$$

Accordingly, given the sequence $\mathcal{U}_{1:F} = \{\mathcal{U}_1, \ldots, \mathcal{U}_F\}$ of scheduled slots and the sequence of generated packets $\mathcal{G}_{1:F} = \{\mathcal{G}_1, \ldots, \mathcal{G}_F\}$, the *URLLC reliability rate* over a window of F frames is the average reliability measure

$$\rho_{\mathrm{U}}\big(\mathcal{U}_{1:F}\big|\mathcal{G}_{1:F}\big) = \tfrac{1}{F}\sum_{f=1}^{F} r(\mathcal{U}_f|\mathcal{G}_f). \tag{5.2}$$

The allocation $\mathcal{U}_{1:F}$ is said to be $(1-\alpha)$-*URLLC reliable* for the generation sequence $\mathcal{G}_{1:F}$ if the following limit holds

$$\lim_{F\to\infty} \rho_{\mathrm{U}}\big(\mathcal{U}_{1:F}\big|\mathcal{G}_{1:F}\big) \geq 1-\alpha. \tag{5.3}$$

This imposes that over a sufficiently long time horizon, the fraction of frames which URLLC packets are served in a timely manner is at least $1-\alpha$.

5.2.3 eMBB efficiency

A scheduler could easily obtain the highest coverage rate of 1 by allocating all S slots to URLLC traffic. However, this would come at the cost of eMBB traffic. The *eMBB efficiency* of an allocation strategy is measured by the fraction of slots available for eMBB transmission over a window of F frames, i.e., as

$$\eta_{\mathrm{e}}(\mathcal{U}_{1:F}) = \tfrac{1}{F}\sum_{f=1}^{F} \tfrac{S-|\mathcal{U}_f|}{S} = 1 - \tfrac{1}{FS}\sum_{f=1}^{F} |\mathcal{U}_f|. \tag{5.4}$$

Since the reliability requirements of URLLC are more stringent, by many orders of magnitude, as compared to eMBB, we focus on meeting URLLC reliability constraints, while serving eMBB traffic is in a best-effort fashion.

5.2.4 URLLC predictor

The scheduler has access to an arbitrary probabilistic URLLC traffic predictor. The predictor may be model-based, e.g., based on a Markov model, or data-driven, e.g., a recurrent neural network, and we make no assumptions on its accuracy. The predictor outputs a probability distribution $q_f(\cdot)$ over all possible subsets of the slot set $\mathcal{S}$. Accordingly, the predictor assigns a probability $q_f(\mathcal{G}_f)$ to each subset $\mathcal{G}_f$ of possible slot indices containing URLLC packets in frame f. This probability generally depends in arbitrary ways on the past observations of the predictor. Such observations include the past decisions $\mathcal{U}_{1:f-1}$ of the scheduler, as well as, possibly partial, information about the previous packet generation subsets $\mathcal{G}_{1:f-1}$. For instance, the predictor may have access to the previous reliability indicators $r(\mathcal{U}_{f'}|\mathcal{G}_{f'})$ with $f' = 1, ..., f-1$ providing information about whether past allocations have been successful or not. Furthermore, while the probability $q_f(\cdot)$ generally ranges over all possible 2^S subsets of slots, practical predictors may, e.g., factorize this distribution so as to reduce complexity [210, 211].

5.3 CP-Based URLLC Resource Allocation

In this section, we introduce the proposed CP-based resource scheduler, proven to satisfy the reliability constraint (5.3) irrespective of the quality of the predictor $q_f(\cdot)$ on which its decisions are based. This important result is obtained by suitably adjusting the number of slots allocated to URLLC traffic, and hence the resulting eMBB efficiency (5.4). We start by reviewing a naïve approach to scheduling that "trusts" the predictor to be accurate and well-calibrated.

5.3.1 Naïve Prediction-Based Scheduler

Assume that the predictor $q_f(\cdot)$ is well-calibrated, in the sense that it provides the actual probability $q_f(\mathcal{G}_f)$ that a certain URLLC traffic pattern $\mathcal{G}_f$ is realized. For model-based predictors, this would be the case if the available domain knowledge is extremely precise; and for data-driven predictors this condition may arise if one has access to large amount of relevant data. Under such ideal conditions, a naïve scheduler would aim at minimizing the number $|\mathcal{U}_f|$ of allocated slots under the constraint that the sum of probabilities $q_f(\mathcal{G})$

Algorithm 4: Greedy Slot Allocation

Input: latency constraint L, set of subsets $\mathbf{\Gamma}$
Output: URLLC slot allocation $\mathcal{U}$

initialize slot allocation $\mathcal{U} = \emptyset$
◁ *Iterate backwards over slots* ▷
for $s = S, S-1, \ldots, 1$ **do**
  if $s \in \cup_{\mathcal{G}\in\mathbf{\Gamma}}\mathcal{G}$ **then**
    ◁ *if slot in any of the predicted sets* ▷
    $\mathcal{U} \leftarrow \mathcal{U} \cup \{s\}$ ◁ *Add the slot s to the allocation* ▷
    for $\mathcal{G} \in \mathbf{\Gamma}$ **do**
      ◁ *Iterate over all predicted sets* ▷
      ◁ *Remove a slot in traffic pattern $\mathcal{G}$ if the corresponding packet can be transmitted in slot s while satisfying the latency condition L* ▷
      $\mathcal{G} \leftarrow \mathcal{G} \setminus \{\max(\{s-L, \ldots, s\} \cap \mathcal{G})\}$
return $\mathcal{U}$

across all arrivals $\mathcal{G}$ that are L-covered by $\mathcal{U}_f$ is no smaller than $1-\alpha$. We propose to address this combinatorial problem through a two-step heuristic approach. First, we find the smallest set $\mathbf{\Gamma}$ of slot generation patterns $\mathcal{G}_f$ to which the predictor $q_f(\cdot)$ assigns a probability at least $1-\alpha$, i.e., we first solve the problem

$$\mathbf{\Gamma}(\alpha|q_f) = \underset{\mathbf{\Gamma}\subseteq 2^{\mathcal{S}}}{\operatorname{argmin}} \quad |\mathbf{\Gamma}| \quad \text{s.t.} \quad \sum_{\mathcal{G}\in\mathbf{\Gamma}} q_f(\mathcal{G}) \geq 1-\alpha. \tag{5.5}$$

This problem can be addressed by sorting the probabilities $q_f(\cdot)$ in decreasing order. Note that, in practice, problem (5.5) can be simplified by restricting the domain, e.g., by considering only traffic patterns of no more than $G_{\max}$ packets.

Once a set $\mathbf{\Gamma}(\alpha|q_f)$ of subsets is identified, the scheduler could find an allocation $\mathcal{U}_f$ that guarantees that, for all patterns $\mathcal{G}_f \in \mathbf{\Gamma}(\alpha|q_f)$, we have $r(\mathcal{U}_f|\mathcal{G}_f) = 1$ and hence all URLLC packets are correctly transmitted within the latency condition. A greedy algorithm satisfying this condition is detailed in Algorithm 4. The approach operates backwards from slot S to slot 1. For any slot s that belongs to any of the traffic patterns in set $\mathbf{\Gamma}$, the slot s is added to the set of allocated slots $\mathcal{U}$. Furthermore, for each pattern $\mathcal{G} \in \mathbf{\Gamma}$, one slot $s' \leq s$ is removed if it is the largest not yet considered and if it is within L time slots of the allocated slot s.

Under suitable ergodicity conditions (see, e.g., [212]), making the strong assumption that the predictor is indeed well-accurate, the reliability inequality (5.3) would be satisfied by the naïve scheduler with probability 1.

5.3.2 CP-Based Scheduler

In practice, one cannot rely on the accuracy of the predictor to guarantee the reliability condition (5.3). Inspired by online CP [83, 84], we now introduce an approach that is guaranteed to meet the condition (5.3) no matter what the accuracy of the predictor is and for every realization of URLLC traffic patterns. While not affecting URLLC reliability, the accuracy of the predictor dictates eMBB efficiency (5.4), with a more accurate predictor yielding a higher eMBB efficiency.

The key idea is to adjust the threshold used in the definition of set (5.5) as a function of the past reliability measures, so as to meet the reliability condition (5.3). Let us define as α_f the target unreliability rate for frame f, which is used in (5.5) to obtain the set $\mathbf{\Gamma}(\alpha_f|q_f)$. A smaller value of α_f yields a larger set $\mathbf{\Gamma}(\alpha_f|q_f)$. Once such a set is identified, the CP-based scheduler applies the same greedy approach as the naïve scheme to identify set $\mathcal{U}_f$ (see Algorithm 4). Intuitively, the target unreliability rate α_{f+1} for frame $f+1$ should be chosen to be small when the average success rate $f^{-1}\sum_{f'=1}^{f} r(\mathcal{U}_{f'}|\mathcal{G}_{f'})$ obtained so far is smaller than $1-\alpha$; and one should increase α_{f+1} if the average success rate so far is larger than $1-\alpha$.

To this end, we assume that at the end of the f-th frame the scheduler gains access to the reliability measure $r(\mathcal{U}_f|\mathcal{G}_f)$. In practice, this requires some minimal feedback from URLLC devices informing the base station of an unsuccessful attempt to transmit a packet. Then, the target per-frame unreliability threshold α_{f+1} is set as $\alpha_{f+1} = \varphi(\theta_{f+1})$, where $\varphi(\cdot)$ is a monotonically increasing function, known as the *stretching function* [84]. The parameter θ_{f+1} is updated as

$$\theta_{f+1} \leftarrow \theta_f + \gamma\Big(r(\mathcal{U}_f|\mathcal{G}_f) - (1-\alpha)\Big), \tag{5.6}$$

where $\gamma > 0$ is an update step. We adopt the stretching function

$$\varphi(\theta) = \tfrac{1}{2}\Big(1 + \sin\Big(\pi\Big(\max\Big\{0, \min\{1,\theta\}\Big\} - 0.5\Big)\Big)\Big), \tag{5.7}$$

which satisfies the conditions in [84, Theorem 1].

By [83, Proposition 4.1], this choice ensures that the difference between the URLLC reliability rate, $\rho_{\mathrm{U}}(\mathcal{U}_{1:F}|\mathcal{G}_{1:F})$, and the target rate $1-\alpha$ satisfies the inequality

$$\Big|\rho_{\mathrm{U}}(\mathcal{U}_{1:F}|\mathcal{G}_{1:F}) - (1-\alpha)\Big| \leq \mathcal{O}(1/F) \tag{5.8}$$

Algorithm 5: CP-Based Scheduler

Input: target unreliability rate $\alpha > 0$, probabilistic predictor $\{q_f\}_{f\in\mathbb{N}}$, latency constraint L, update step $\gamma > 0$
Output: URLLC slot allocations $\mathcal{U}_1, \mathcal{U}_2, \ldots$

initialize threshold $\theta_1 \leftarrow \varphi^{-1}(\alpha)$
for $f = 1, \ldots,$ **do**
  retrieve previous pattern $\mathcal{G}_{f-1}$
  find $\mathcal{U}_f$ using Algorithm 4 for $\mathbf{\Gamma}(\varphi(\theta_f)|q_f)$ using (5.5)
  allocate slots $\mathcal{U}_f$ for URLLC traffic $\mathcal{G}_f$ in frame f
  obtain reliability indicator $r(\mathcal{U}_f|\mathcal{G}_f)$ via (5.1)
  ◁ *Adapt calibration parameter* ▷
  $\theta_{f+1} \leftarrow \theta_f + \gamma(r(\mathcal{U}_f|\mathcal{G}_f) - (1-\alpha))$
return $\mathcal{U}_1, \mathcal{U}_2, \ldots$

for any number of frames, F, and irrespective of the specific realized sequence of traffic patterns. This condition yields the limit (5.3) as the number of frames, F, grows large.

5.4 Experiments and Conclusions

To validate the proposed approach, we conducted experiments under a Markov packet generation mechanism. Recall that the proposed scheme provides guarantees that do not depend on the statistics of the packet arrival process. The arrival process is defined by four parameters $(p^-, p^+, G_{\min}, G_{\max})$. Accordingly, given the current traffic pattern $\mathcal{G}_f$, the next traffic pattern $\mathcal{G}_{f+1}$ has a number of packets equal to $G_{f+1} = [G_f + W_{f+1}]_{G_{\min}}^{G_{\max}}$, where W_f is a ternary variable that equals $W_{f+1} = 1$ with probability p^+, $W_{f+1} = -1$ with probability p^-, and $W_{f+1} = 0$ otherwise. The function $[\cdot]_{G_{\min}}^{G_{\max}}$ clips the input argument within the range $[G_{\min}, G_{\max}]$. Given a number $G_{f+1} \neq G_f$ of packets, the traffic pattern $\mathcal{G}_{f+1}$ is selected uniformly among all subsets of cardinality G_{f+1} that can be obtained from pattern $\mathcal{G}_f$ by adding a slot (if $G_{f+1} > G_f$) or removing a slot (if $G_{f+1} < G_f$). Otherwise, if $G_{f+1} = G_f$, we set $\mathcal{G}_{f+1} = \mathcal{G}_f$. While simplistic, this mechanism allows us to draw insightful conclusions on the role of predictors in the performance of schedulers.

To this end, we assume that the predictor $q_f(\cdot)$ adopts the same Markov model of the ground-truth packet generation mechanism, but with generally mismatched probabilities $\hat{p}^+$ and $\hat{p}^-$ in lieu of the true probabilities p^+ and p^-.

Fig. 5.1 shows the generated packets $\{\mathcal{G}_f\}$ over the last 200 frames of a 2000 frames run, along with the allocation $\{\mathcal{U}_f\}$ and reliability indicators (5.1) in the side bars. Each frame consists of $S = 12$ slots, the URLLC latency is $L = 1$, the learning rate $\gamma = 0.1$,

and traffic follows $G^{\min} = 0$ and $G^{\max} = 6$ and $p^+ = p^- = 0.16$. We consider two predictors: The first underestimates the parameters with $\hat{p}^+ = \hat{p}^- = 0.02$, while the second overestimates $\hat{p}^+ = \hat{p}^- = 0.40$. The conventional scheduler either fails to meet (5.3) using the underestimating predictor (covering 82% instead of $1 - \alpha = 90\%$), or allocates an excessively large number of slots using the overestimating predictor. In contrast, the CP-based predictor can effectively adjust the eMBB efficiency to the quality of the predictor, always meeting the reliability constraint (5.3). For example, it trades excessive coverage (98% to 90%) into higher eMBB efficiency (45% to 66% as in Fig. 5.1(c)).

We now set $\alpha = 0.01$ and $\gamma = 0.05$, and investigate the impact of a mismatch between the URLLC traffic model assumed by the predictor and the ground-truth model. We set $p^+ = p^- = p$ and $\hat{p}^+ = \hat{p}^- = \hat{p}$, and let both parameters vary. Fig. 5.3 shows the empirical URLLC reliability rate (5.2) and the empirical eMBB efficiency (5.4) at the completion of $F = 4000$ frames for both the naïve scheduler and the CP-based scheduler. The naïve scheduler is significantly affected by a mismatch between predictor and ground-truth packet generation mechanism, yielding either ill empirical coverage (below $1 - \alpha = 0.99$) or over coverage. In contrast, the CP-based predictor is able to flatten the coverage to asymptotically reach the long-term target $1 - \alpha$.

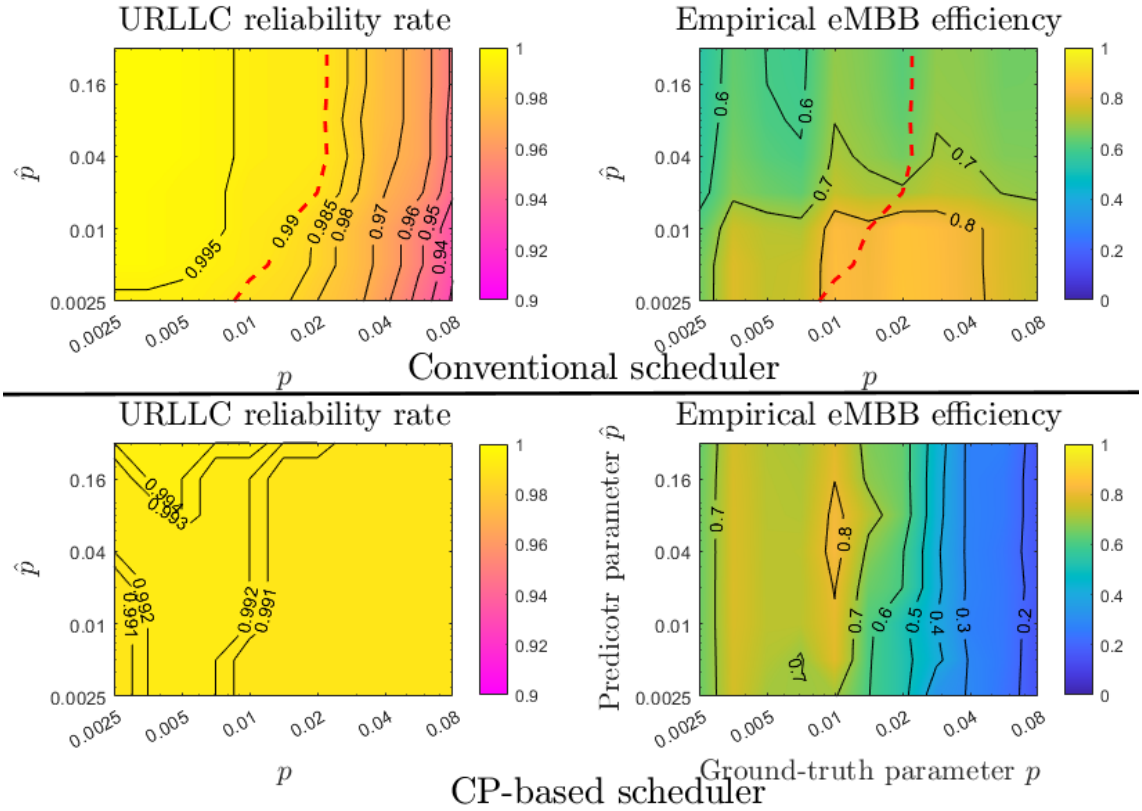


Fig. 5.3 URLLC reliability rate (5.2) and eMBB efficiency (5.4) for conventional scheduler (Sec. 5.3.1) and CP-based scheduler (Sec. 5.3.2) as a function of the ground-truth traffic parameter and predictor parameter. The target rate is $1 - \alpha = 0.99$ (dashed red line for conventional scheduler; the CP-based scheduler always satisfies the reliability condition).

Chapter 6

Cross-Validation-Based Conformal Risk Control

6.1 Introduction

6.1.1 Context and Motivation

One of the key requirements for the application of artificial intelligence (AI) tools to risk-sensitive fields such as healthcare and engineering is the capacity of AI algorithms to quantify their uncertainty [213, 214]. This requires guarantees on the adherence of the "error bars" produced by the AI model to the true predictive uncertainty. The predictive uncertainty encompasses both the epistemic uncertainty caused by limited availability of data and the aleatoric uncertainty inherent in the randomness of data generation [28]. Without making strong assumptions on the data generation mechanism it is generally impossible to provide strict uncertainty quantification guarantees for any input, but assumption-free guarantees can be established on average over validation and test data [215]. *Conformal prediction* (CP) [17, 19], and its extension *conformal risk control* (CRC) [91], are widely established methodologies for the evaluation of predictors with provable uncertainty quantification properties.

To elaborate, assume access to a data set $\mathcal{D}$ of N pairs of examples consisting of input x and output y. Based on the data set $\mathcal{D}$ and on a class of point predictors, CP and CRC produce a set predictor $\Gamma(x|\mathcal{D})$ mapping a test input x into a subset of the output space. The size of the set predictor $\Gamma(x|\mathcal{D})$ provides a measure of the uncertainty of the predictor for input x [19]. On average over the data set $\mathcal{D}$ and over a test input-output pair (x, y),

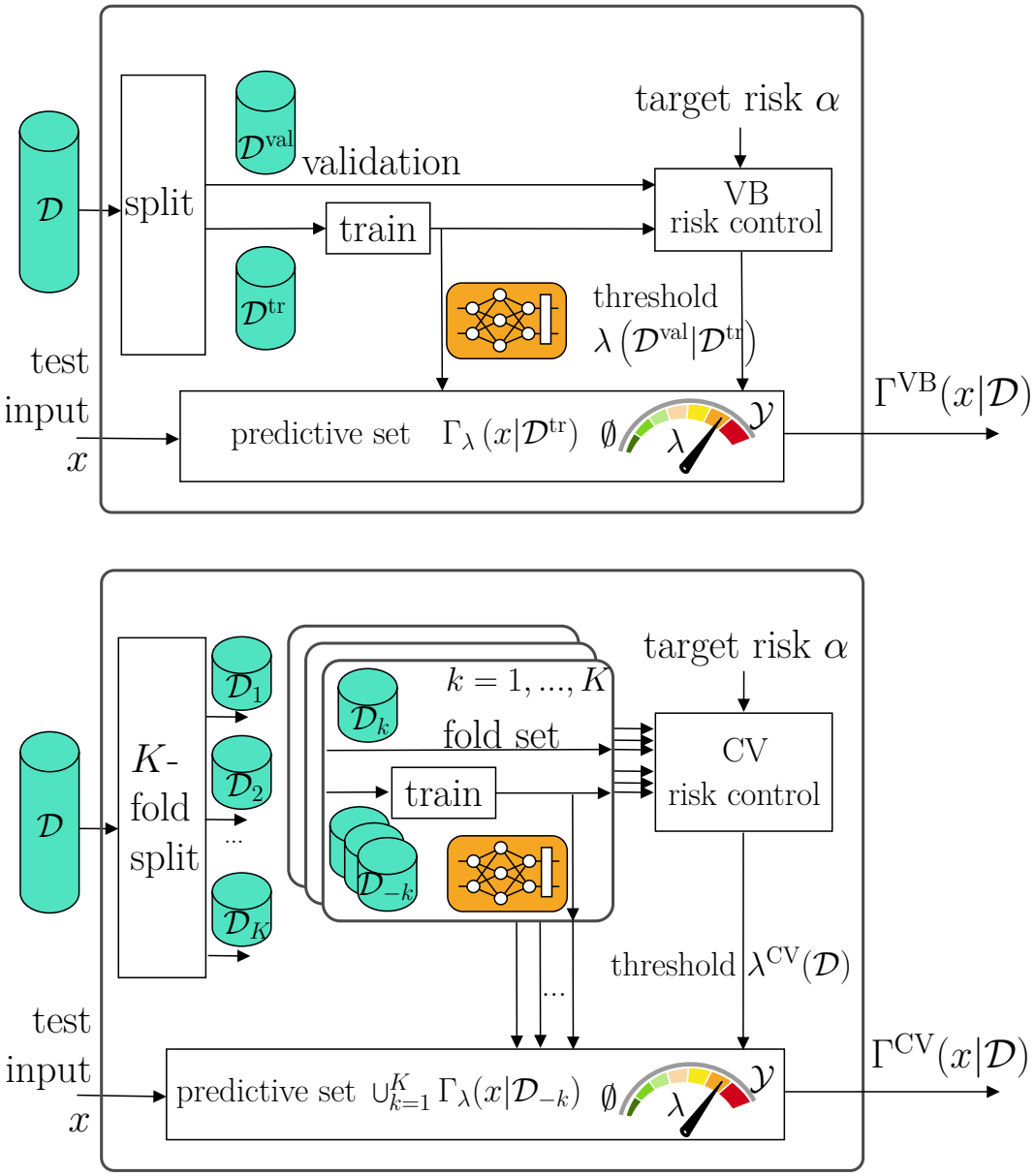


Fig. 6.1 Illustration of **(top)** the existing validation-based conformal risk control (VB-CRC) [91]; and **(bottom)** the proposed method cross-validation-based conformal risk control (CV-CRC), which aims at reducing the predictive sets sizes by reusing the available data $\mathcal{D}$ more efficiently.

we wish to guarantee the calibration condition

$$\mathrm{E}_{\boldsymbol{\mathcal{D}},\mathbf{x},\mathbf{y}\sim p_0(\mathcal{D},x,y)}\Big[\ell\Big(\mathbf{y},\Gamma(\mathbf{x}|\boldsymbol{\mathcal{D}})\Big)\Big] \leq \alpha, \tag{6.1}$$

where boldface fonts denote random quantities, $\ell(\cdot,\cdot)$ is a loss measure, and α a user-specified maximum average loss level. In (6.1), under the joint distribution $p_0(\mathcal{D},x,y)$, the examples in the data set $\boldsymbol{\mathcal{D}}$ and the test pair $(\mathbf{x},\mathbf{y})$ are assumed to be independent identically distributed (i.i.d.), or, more generally *exchangeable*.

CRC can satisfy the requirement (6.1) for any user-specified target average loss level α, as long as the loss function is bounded and it decreases as the predicted set grows. Examples of such loss functions are the 0-1 miscoverage probability

$$\ell(y,\Gamma) = \mathbb{1}(y \notin \Gamma), \tag{6.2}$$

which returns 1 if the true label y is not in the set Γ and 0 otherwise, and the *false negative rate*, which returns the fraction of true values of y that are not included in set Γ for multi-label problems [91].

The requirement (6.1) can be always satisfied for such monotonic loss functions by returning as set predictor Γ the entire set of possible values for the output variable y. However, a set predictor is useful only as long as it is of moderate average size. The motivation of this work is to construct a set predictor that meets (6.1), while producing small predictive sets even in the presence of a limited data set $\mathcal{D}$.

6.1.2 State of the Art

CP addresses the design of set predictors satisfying the calibration condition (6.1) in the special case of the miscoverage loss (6.2) [17–19]. There are several variants of CP, including validation-based CP (VB-CP), *cross-validation-based CP (CV-CP)* [90], and full CP [17]. While full CP is considered to be impractical, requiring many rounds of retraining, VB-CP splits the data set into training and validation data sets, and it operates over a single round of training. However, the need to devote a separate data set for validation can significantly reduce the quality of the trained model, resulting in predictive sets of large sizes when data are limited [90].

CV-CP reduces the computational complexity as compared to full CP, while reducing the predicted set size as compared to VB-CP. This is done by partitioning the available data set into multiple folds, each acting as a validation data set for the model trained based on leave-fold-out data. At the cost of increasing the complexity, requiring as many training

rounds as the number of folds, CV-CP was shown to produce important savings in terms of prediction set sizes [111, 216, 217].

Other extensions of CP include CP-aware training strategies [76, 77], prediction under distributional shifts [78], improvements in the training algorithms [79, 80], novel calibration metrics [81, 82], applications to engineering problems [111, 218], and online versions [83, 84] with applications [219, 113].

CRC generalizes CP to address the calibration criterion (6.1) for a wider class of risks, with the only constraints that the risk function be bounded and monotonic in the predicted set size [91, 92, 94, 84]. The original CRC is validation-based, and hence it may be referred to as VB-CRC for consistency with the terminology applied above for CP. Accordingly, it relies on a split of the data set into training and validation sets, resulting in inefficient predictive sets when data are limited.

6.1.3 Main Contributions

In this chapter, we introduce a novel version of CRC based on cross-validation. The proposed CV-CRC method generalizes CV-CP, supporting arbitrary bounded and monotonic risk functions. As we will demonstrate, the design and analysis of CV-CRC are non-trivial extensions of CV-CP, requiring new definitions and proof techniques.

The rest of the chapter is organized as follows. Sec. 6.2 provides the necessary background, while CV-CRC is presented in Sec. 6.3. Numerical experiments are reported in Sec. 6.4, and Sec. 6.5 draws some conclusions. All proofs are deferred to the supplementary material.

6.2 Background

Consider $N+1$ data points

$$\underbrace{(\mathbf{x}[1],\mathbf{y}[1])}_{=\mathbf{z}[1]}\ ,\ \underbrace{(\mathbf{x}[2],\mathbf{y}[2])}_{=\mathbf{z}[2]}\ ,\ \dots\ ,\ \underbrace{\mathbf{x}[N+1],\mathbf{y}[N+1])}_{=\mathbf{z}[N+1]} \tag{6.3}$$

over the sample space $\mathcal{X}\times\mathcal{Y}$ that are drawn according to an *exchangeable* joint distribution $p_0(\mathcal{D},x,y)$ over index $i=1,\dots,N$. The first N data points constitute the data set $\mathcal{D}=\{z[i]=(x[i],y[i])\}_{i=1}^{N}$, while the last data point $z[N+1]$ is the test pair, which is also denoted as $z=(x,y)$. We fix a loss function $\ell:\mathcal{Y}\times 2^{\mathcal{Y}}\to\mathbb{R}$, which, given any label $y\in\mathcal{Y}$ and a predictive set $\Gamma\subseteq\mathcal{Y}$, returns a loss bounded as

$$b\le\ell(y,\Gamma)\le B \tag{6.4}$$

for some constants $B < \infty$ and $b \in \{-\infty\} \cup \mathbb{R}$. We further require that the loss is monotonic in the predictive set Γ in the sense that the following implication holds

$$\Gamma_1 \subseteq \Gamma_2 \quad \Rightarrow \quad \ell(y, \Gamma_1) \geq \ell(y, \Gamma_2) \quad \text{for each } y \in \mathcal{Y}. \tag{6.5}$$

Note that the 0-1 miscoverage loss (6.2) assumed by CP satisfies (6.4) with $b = 0$ and $B = 1$, and it also satisfies the implication (6.5).

For a given data set $\mathcal{D}$, VB-CRC uses a two-step procedure to satisfy the constraint (6.1) for some target average loss α in the interval

$$b \ \leq \ \alpha \ \leq \ B. \tag{6.6}$$

To start, as illustrated in the top panel of Fig. 6.1, the available data set $\mathcal{D}$ is split into N^{tr} examples forming the *training set* $\mathcal{D}^{\text{tr}}$ and $N^{\text{val}} = N - N^{\text{tr}}$ points forming the *validation set* $\mathcal{D}^{\text{val}}$ with $\mathcal{D} = \mathcal{D}^{\text{tr}} \cup \mathcal{D}^{\text{val}}$. In the first step of VB-CRC, a model is trained based on the training set $\mathcal{D}^{\text{tr}}$ using any arbitrary scheme. Then, in the second step, VB-CRC determines a threshold $\lambda \in \mathbb{R}$ by using the validation data set $\mathcal{D}^{\text{val}}$. As explained next, the threshold λ dictates which labels $y \in \mathcal{Y}$ are to be included in the prediction set $\Gamma_\lambda(x|\mathcal{D}^{\text{tr}})$ for any test input x as follows.

A *nonconformity (NC) score* $\text{NC}((x, y)|\mathcal{D}^{\text{tr}})$ is selected that evaluates the loss of the trained predictor on a pair (x, y). Examples of NC scores include the residual between the label and a trained predictor for regression problems and the log-loss for classification problems [220, 221, 19]. With the given NC score, the set prediction is obtained as

$$\Gamma_\lambda(x|\mathcal{D}^{\text{tr}}) = \Big\{ y' \in \mathcal{Y} \Big| \text{NC}((x, y')|\mathcal{D}^{\text{tr}}) \leq \lambda \Big\}, \tag{6.7}$$

thus including all labels $y' \in \mathcal{Y}$ with NC score smaller or equal to the threshold λ. By design, the set (6.7) satisfies the nesting property

$$\lambda_1 < \lambda_2 \quad \Rightarrow \quad \Gamma_{\lambda_1}(x|\mathcal{D}^{\text{tr}}) \subseteq \Gamma_{\lambda_2}(x|\mathcal{D}^{\text{tr}}) \tag{6.8}$$

for any input x and data sets $\mathcal{D}^{\text{tr}}$.

We define the *risk* as the population, or test, loss of the predicted set (6.7) as

$$R(\lambda|\mathcal{D}^{\text{tr}}) = \text{E}_{\mathbf{x},\mathbf{y} \sim p_0(x,y)} \Big[\ell\Big(\mathbf{y}, \Gamma_\lambda(\mathbf{x}|\mathcal{D}^{\text{tr}})\Big) \Big]. \tag{6.9}$$

Given the validation data set $\mathcal{D}^{\text{val}} = \{(x^{\text{val}}[i], y^{\text{val}}[i])\}_{i=1}^{N^{\text{val}}}$, the risk (6.9) can be estimated as

$$\hat{R}^{\text{val}}(\lambda|\mathcal{D}^{\text{tr}}, \mathcal{D}^{\text{val}}) = \frac{1}{N^{\text{val}}+1}\left(\sum_{i=1}^{N^{\text{val}}} \ell\Big(y^{\text{val}}[i], \Gamma_\lambda(x^{\text{val}}[i]|\mathcal{D}^{\text{tr}})\Big) + B\right), \tag{6.10}$$

which is a function of the threshold λ. This corresponds to a regularized, biased, empirical estimate of the risk (6.9) that effectively adds an $(N+1)$-th dummy validation example with maximal loss B.

VB-CRC chooses the lowest threshold λ such that the estimate (6.10) is no larger than the target average risk α as in

$$\lambda^{\text{VB}}(\mathcal{D}^{\text{val}}|\mathcal{D}^{\text{tr}}) = \inf_{\lambda} \left\{ \lambda \middle| \hat{R}^{\text{val}}(\lambda|\mathcal{D}^{\text{tr}}, \mathcal{D}^{\text{val}}) \leq \alpha \right\}. \tag{6.11}$$

With this threshold choice, as proven in [91], the set predictor (6.7) obtained via VB-CRC, i.e.,

$$\Gamma^{\text{VB}}(x|\mathcal{D}^{\text{tr}}, \mathcal{D}^{\text{val}}) = \Gamma_{\lambda^{\text{VB}}(\mathcal{D}^{\text{val}}|\mathcal{D}^{\text{tr}})}(x|\mathcal{D}^{\text{tr}}) \tag{6.12}$$

ensures the desired condition (6.1). More precisely, the condition (6.1) holds for any fixed training set $\mathcal{D}^{\text{tr}}$, i.e., we have the inequality

$$\text{E}_{\boldsymbol{\mathcal{D}}^{\text{val}}, \mathbf{x}, \mathbf{y} \sim p_0(\mathcal{D}^{\text{val}}, x, y)} \Big[\ell\Big(\mathbf{y}, \Gamma^{\text{VB}}(\mathbf{x}|\mathcal{D}^{\text{tr}}, \boldsymbol{\mathcal{D}}^{\text{val}})\Big)\Big] \leq \alpha. \tag{6.13}$$

Furthermore, in order for (6.13) to hold, VB-CRC only requires the validation data $\mathcal{D}^{\text{val}}$ and test pair (x, y) to be exchangeable.

6.3 Cross-Validation Conformal Risk Control

While VB-CRC reviewed in the previous section guarantees the average risk condition (6.13), splitting the available data set into training and validation sets may potentially lead to inefficient set predictors, having large predictive sets on average. In this section, we introduce the proposed CV-CRC scheme that aims at improving the efficiency of VB-CRC [91] via cross-validation [90], while still guaranteeing condition (6.1).

To start, as illustrated in the bottom panel of Fig. 6.1, the available data set $\mathcal{D} = \{z[i]\}_{i=1}^{N}$ is partitioned using a fixed mapping into K *folds* $\mathcal{D} = \{\mathcal{D}_k\}_{k=1}^{K}$ of N/K-samples each, which is assumed to be an integer. We will write each k-th fold as

$$\mathcal{D}_k = \{(x_k[1], y_k[1]), \ldots, (x_k[N/K], y_k[N/K])\}, \tag{6.14}$$

and we will denote the mapping of the i-th data point $z[i]$ to its fold index as $k[i] : \{1, \dots, N\} \to \{1, \dots, K\}$. Like VB-CRC, CV-CRC operates in two steps.

In the first step, for any k-th fold, a model is trained using the leave-fold-out training set $\mathcal{D}_{-k} = \mathcal{D} \setminus \mathcal{D}_k$ of $N - N/K$ samples. Accordingly, unlike VB-CRC, K training rounds are required for CV-CRC. In the second step, as we will detail, CV-CRC determines a threshold λ to determine which values of the output y to include in the predicted set.

Given a threshold λ, CV-CRC produces the predictive set

$$\Gamma_\lambda^{\mathrm{CV}}(x|\mathcal{D}) = \Big\{ y' \in \mathcal{Y} \Big| \min_{k \in \{1,\dots,K\}} \Big\{ \mathrm{NC}((x, y')|\mathcal{D}_{-k}) \Big\} \leq \lambda \Big\}, \tag{6.15}$$

which includes all labels $y' \in \mathcal{Y}$ with minimum, i.e., best case, NC score across the K folds, that is not larger than λ.

To determine the threshold λ, CV-CRC estimates the population risk (6.9) using cross-validation as

$$\hat{R}^{\mathrm{CV}}(\lambda|\mathcal{D}) = \tfrac{1}{K+1} \Bigg(\sum_{k=1}^{K} \tfrac{K}{N} \sum_{j=1}^{N/K} \ell\Big(y_k[j], \Gamma_\lambda\Big(x_k[j] \Big| \mathcal{D}_{-k} \Big) \Big) + B \Bigg). \tag{6.16}$$

The cross-validation-based estimate (6.16) can be interpreted as the conventional cross-validation loss evaluated on an augmented data set

$$\mathcal{D}^{\mathrm{aug}} = \Big\{ \underbrace{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_K}_{=\mathcal{D}}, \mathcal{D}_{\mathrm{dummy}} \Big\}, \tag{6.17}$$

with the first K folds being the available data set $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_K\}$, and the additional $(K+1)$-th fold containing N/K dummy points with the maximal loss of B. In a manner similar to VB-CRC, the addition to dummy data points acts as a regularizer for the estimate (6.16), which is required to provide performance guarantees.

Finally, CV-CRC selects the threshold λ by imposing that the cross-validation based estimate (6.16) of the loss is no larger than the target average loss value α as in

$$\lambda^{\mathrm{CV}}(\mathcal{D}) = \inf_{\lambda} \Big\{ \lambda \Big| \hat{R}^{\mathrm{CV}}(\lambda|\mathcal{D}) \leq \alpha \Big\}. \tag{6.18}$$

CV-CRC reduces to the jackknife-minmax scheme in [90] when evaluated with the miscoverage loss (6.2) in the special case of $K = N$ folds.

Theorem 1. *Fix any bounded and monotonic loss function $\ell(\cdot, \cdot)$ satisfying conditions* (6.4) *and* (6.5)*, and any NC score* $\mathrm{NC}((x, y)|\mathcal{D}^{tr})$ *that is permutation-invariant with respect to*

the ordering of the examples in the training set $\mathcal{D}^{tr}$. For any number of folds satisfying $K \geq B/(\alpha - b) - 1$, the CV-CRC predictive set $\Gamma^{CV}_{\lambda^{CV}(\mathcal{D})}(x|\mathcal{D})$ with (6.15) and (6.18) *guarantees the condition*

$$\mathrm{E}_{\boldsymbol{\mathcal{D}},\mathbf{x},\mathbf{y}\sim p_0(\mathcal{D},x,y)}\Big[\ell\Big(\mathbf{y}, \Gamma^{CV}(\mathbf{x}|\boldsymbol{\mathcal{D}})\Big)\Big] \leq \alpha. \tag{6.19}$$

The theorem thus confirms that CV-CRC meets the desired condition (6.1). In this regard, we note that, as in (6.1), the average loss in (6.19) includes averaging over the entire data set $\mathcal{D}$, unlike the condition (6.13) satisfied by VB-CRC. Furthermore, Theorem 1 requires the NC score to be permutation-invariant with respect to the data points in the training set, which is not the case for VB-CRC. Permutation-invariance is also needed for CV-CP [90], as well as for full CP [17]. In practice, a permutation-invariant NC score can be obtained by implementing permutation-invariant training schemes such as full gradient descent, in which the final trained model does not depend on the ordering of the training data points.

6.4 Examples

In this section, we numerically validate the proposed CV-CRC using two synthetic examples. The first is a vector regression problem, whereas the second concerns the problem of temporal point process prediction [222, 223].

6.4.1 Vector Regression

Inspired by the example in [90], we first investigate a vector regression problem in which the output variable $y = [y_1, \ldots, y_m]^\top$ is m-dimensional. The joint distribution of data set $\mathcal{D}$ and test pair (x, y) is obtained as

$$p_0(\mathcal{D}, x, y) = \int p_0(\phi) \left(\prod_{i=1}^{N+1} p_0(x[i]) p_0(y[i]|x[i], \phi) \right) \mathrm{d}\phi, \tag{6.20}$$

where $(x[N+1] = x, y[N+1] = y)$ is the test example, and we have the Gaussian distributions

$$p_0(x) = \mathcal{N}(x|0, d^{-1} I_d), \tag{6.21a}$$

$$p_0(y|x, \phi) = \mathcal{N}(y|\phi^\top \cdot x, \beta_0^{-1} I_m), \tag{6.21b}$$

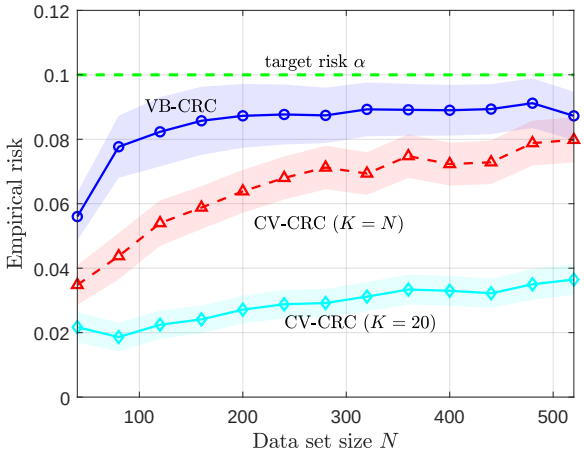


Fig. 6.2 Empirical risk of VB-CRC and CV-CRC for the vector regression problem.

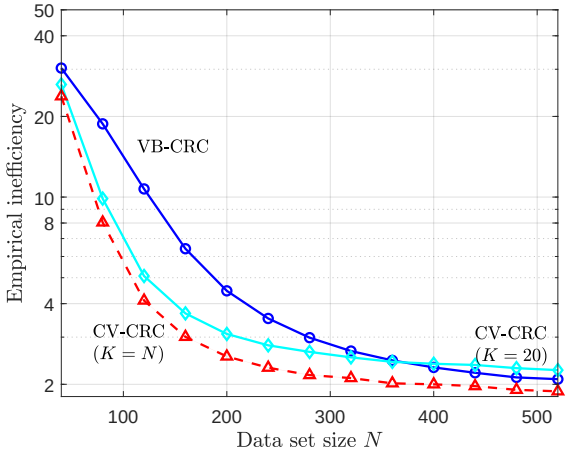


Fig. 6.3 Empirical inefficiency of VB-CRC and CV-CRC for the vector regression problem.

while $p_0(\phi)$ is a mixture of Gaussians with means determined by an i.i.d. Bernoulli vector $\mathbf{b}$ as

$$p_0(\phi) = \mathrm{E}_{\mathbf{b} \overset{\text{i.i.d.}}{\sim} \mathrm{Bern}(0.5)} \left[\mathcal{N}(\phi | \mu_0 \mathbf{b}, \gamma_0^{-1} I_d) \right]. \tag{6.22}$$

We set $\mu_0 = 10$, $\gamma_0 = 1$, $\beta_0 = 4$, $d = |\mathcal{X}| = 50$, and $m = |\mathcal{Y}| = 30$. Note that the distribution (6.20) is exchangeable.

Using maximum-likelihood learning, given a training data set $\mathcal{D}^{\mathrm{tr}}$, we obtain the model parameter $\phi^{\mathrm{ML}}_{\mathcal{D}^{\mathrm{tr}}}$ used for the linear prediction model $\hat{y}(x|\mathcal{D}^{\mathrm{tr}}) = (\phi^{\mathrm{ML}}_{\mathcal{D}})^\top x$ as $\phi^{\mathrm{ML}}_{\mathcal{D}^{\mathrm{tr}}} = X^\dagger_{\mathcal{D}^{\mathrm{tr}}} Y_{\mathcal{D}^{\mathrm{tr}}}$, where $(\cdot)^\dagger$ denotes the pseudo-inverse, $(\cdot)^\top$ denotes transpose, and the input and label data matrices $X_{\mathcal{D}} \in \mathbb{R}^{N \times d}$ and $Y_{\mathcal{D}} \in \mathbb{R}^{N \times m}$ have input $(x^{\mathrm{tr}}[i])^\top$ and label $(y^{\mathrm{tr}}[i])^\top$ as their ith rows, respectively.

The NC score is set to the maximum prediction residual across the m dimensions of the output variable y as

$$\mathrm{NC}((x,y)|\mathcal{D}^{\mathrm{tr}}) = 2\Big\|y - \hat{y}(x|\mathcal{D}^{\mathrm{tr}})\Big\|_{\infty}, \tag{6.23}$$

where the infinity norm $\left\|\cdot\right\|_{\infty}$ returns the largest magnitude of its input vector. This results in predictive sets (6.12) and (6.15) with (6.18) in the form of $\Gamma = \Gamma_1 \times \cdots \times \Gamma_m$, with $\times$ being the Cartesian product and

$$\Gamma_j^{\mathrm{VB}} = \left\{ y_j \,\middle|\, |y_j - [\hat{y}(x|\mathcal{D}^{\mathrm{tr}})]_j| \leq \lambda^{\mathrm{VB}}(\mathcal{D}^{\mathrm{val}}|\mathcal{D}^{\mathrm{tr}})/2 \right\} \tag{6.24}$$

with $[\cdot]_j$ standing for the jth element of its argument for VB-CRC, and

$$\Gamma_j^{\mathrm{CV}} = \bigcup_{k=1}^{K} \left\{ y_j \,\middle|\, |y_j - [\hat{y}(x|\mathcal{D}_{-k})]_j| \leq \lambda^{\mathrm{CV}}(\mathcal{D})/2 \right\} \tag{6.25}$$

for CV-CRC. The loss function used in the risk (6.1) is defined as

$$\ell(y,\Gamma) = \frac{1}{m}\sum_{j=1}^{m} \mathbb{1}\Big(y_j \notin \Gamma_j\Big), \tag{6.26}$$

which evaluates the fraction of entries of vector y that are not included in the predictive set. This loss satisfies condition (6.4) with $b = 0$ and $B = 1$. Note that CP is not applicable to this loss, since it is different from (6.2).

Lastly, we define the *inefficiency* as the size of the predictive set evaluated as the average over all dimensions of the predictive intervals across the m dimensions of the output y, i.e., $\mathrm{ineff}(\Gamma) = \frac{1}{m}\sum_{j=1}^{m} \left|\Gamma_j\right|$.

For target risk $\alpha = 0.1$, the empirical risk and empirical inefficiency of $N^{\mathrm{te}} = 200$ test covariate-output pairs, averaged over 50 independent simulations, are shown in Fig. 6.2 and Fig. 6.3. Fig. 6.2, validates the theoretical result that CRC schemes satisfy condition (6.1). However, from Fig. 6.3, VB-CRC is observed to have a larger inefficiency than CV-CRC, particularly in the small data set size regime. Thus, CV-CRC uses data more efficiently, with $K = 20$ folds striking a good balance between inefficiency and computational complexity in this regime.

6.4.2 Temporal Point Process Prediction

A temporal process consists of a sequence of events at random times $t_1, t_2, \ldots$ with $t_1 < t_2 < \ldots$ As illustrated in Fig. 6.4, given the past d events' timings $x = \{t_1, \ldots, t_d\}$, the goal

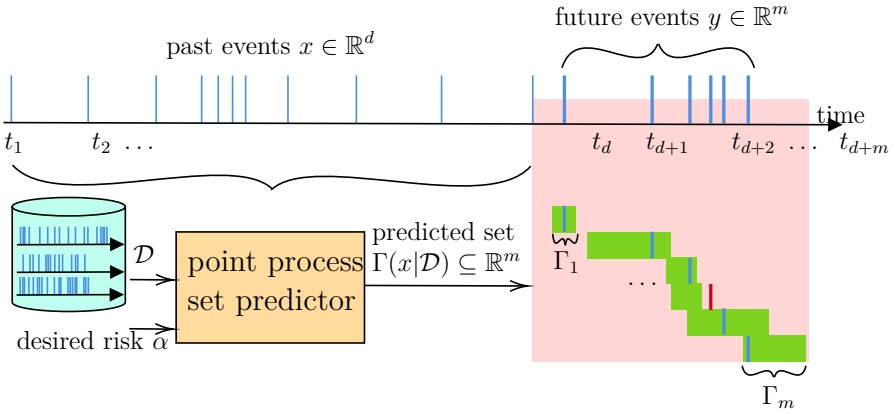


Fig. 6.4 Temporal point process prediction: After observing the past d times $t_1, \ldots, t_n$, a point process set predictor outputs predictive intervals $\Gamma_j(x|\mathcal{D})$ for each of the next m points with $j = 1, \ldots, m$.

is to output intervals $\Gamma_j(x|\mathcal{D})$ for each of the following m events with $j = 1, \ldots, m$. The loss function is defined as in (6.26).

Data and test sequences of timings are generated following a self-exciting Hawkes process [224] with intensity function

$$\lambda(t|\mathcal{H}_t) = \mu + \sum_{i:t_i<t} \left(\alpha_1 \beta_1 e^{-\beta_1 (t-t_i)} + \alpha_2 \beta_2 e^{-\beta_2 (t-t_i)} \right),$$

with $\mu = 0.2, \alpha_1 = \alpha_2 = 0.4, \beta_1 = 1$ and $\beta_2 = 20$ [222]. The predictor is a recurrent neural network that outputs a predictive density function $p(t_{i+1}|t_1, \ldots, t_i, \phi_{\mathcal{D}^{\text{tr}}})$ with trained parameter $\phi_{\mathcal{D}^{\text{tr}}}$ [222]. The median $\hat{t}_{i+1}(t_1, \ldots, t_i, \phi_{\mathcal{D}^{\text{tr}}})$ of the predictive distribution is used as the point estimate for the $(i+1)$-th event. For $i > d$, estimates $\{\hat{t}_j\}_{j=d+1}^{i-1}$ are used in lieu of the correct timings in the point prediction.

VB-CRC (6.12) produces intervals

$$\Gamma_j^{\text{VB}} = \left\{ y_j \,\middle|\, |y_j - \hat{t}_{d+j}(\mathcal{D}^{\text{tr}})| \leq \gamma^j \lambda^{\text{VB}}(\mathcal{D}^{\text{val}}|\mathcal{D}^{\text{tr}})/2 \right\}, \tag{6.27}$$

where multiplication by the interval common ratio $\gamma = 1.2$ increases the interval sizes for later predictions, and for the CV-CRC (6.15), we have

$$\Gamma_j^{\text{CV}} = \bigcup_{k=1}^{K} \left\{ y_j \,\middle|\, |y_j - \hat{t}_{d+j}(\mathcal{D}_{-k})| \leq \gamma^j \lambda^{\text{CV}}(\mathcal{D})/2 \right\}. \tag{6.28}$$

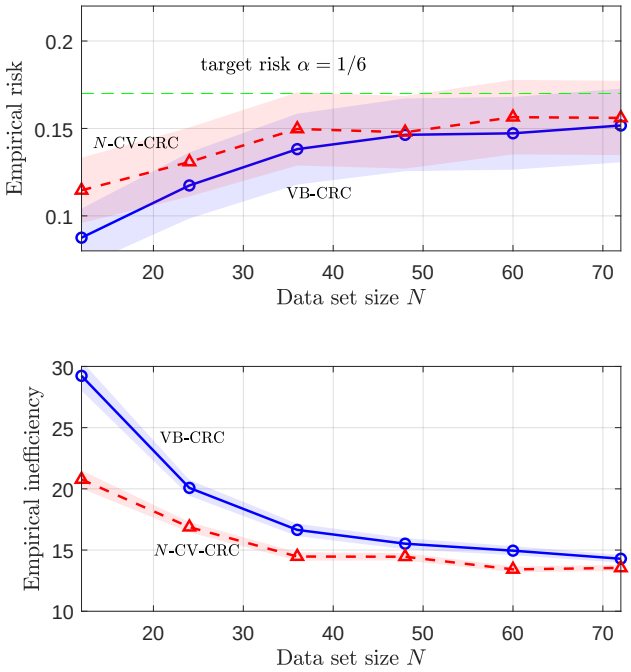


Fig. 6.5 Empirical risk (top) and inefficiency (bottom) of VB-CRC and N-CV-CRC for the temporal point process prediction problem.

We set the length of the observed sequence as $d = 60$, and predict the next $m = 6$ events. We allow one event on average to lie outside the predicted intervals, i.e., $\alpha = 1/6$. We average over 200 independent simulations with $N^{\text{te}} = 1000$ test points in each run.

The top panel of Fig. 6.5 illustrate the test risk (6.26) as function of data set size N, validating that both scheme attain risks lower than the desired level α. The bottom panel of the figure shows that CV-CRC with $K = N$ reduces the average size of the predicted intervals.

6.5 Conclusion

In this chapter, we have introduced a novel conformal risk control (CRC) scheme based on cross-validation, generalizing cross-validation CP to losses beyond miscoverage. The proposed CV-CRC was shown to provably control the average risk, with experiments demonstrating it to be more efficient than VB-CRC when the available data for training and calibration are scarce. Further work may consider using the jackknife+ of [90] instead

of the jackknife-minmax for more efficient predictive sets; and extending the scheme to meta-learning [116].

Chapter 7

Conclusions

7.1 Main Conclusions and Thesis Achievements

In this dissertation, we focused on ways to enhance the reliability and sample-efficiency of AI models. Both of these performance metrics go beyond the well-studied "accuracy" that characterizes many of the early deep-learning solutions. Reliable prediction means the models quantify better the uncertainty associated with their predictions, in light of the amount and test-data similarity to the training data. Sample-efficiency means that predictors are trained using small amount of labeled test-data. Numerical experiments of various communication systems predictors validated the theoretical claims.

We now restate the research questions as in Section 1.2 and list the main conclusions drawn while addressing each one.

- **Research Question 1: Is it possible to enhance the reliability of learning-based communication algorithms in the challenging regime of few pilot symbols?**
  - It is possible to enhance the reliability of predictors, by applying Bayesian learning that leverages disagreements among multiple models, projecting uncertainties regarding the model into the confidence level of the predictions.
  - For the symbol demodulation problem, we show that training a Bayesian neural network via ensembling, results in a more accurate and better-calibrated demodulators.
  - By applying meta-learning, we show that a symbol demodulator can adapt quickly, meaning few pilots are needed for the test frame, when past frames with different channel conditions is available.

  - When integrating Bayesian learning and meta-learning, we can enjoy the merits of reliability and sample-efficiency, on the expense of increased computational complexity.
  - The ability to demodulate using few pilots can be utilized by allocating less symbols of the frame into pilots and freeing them for data symbols, hence increasing the net spectral efficiency.

- **Research Question 2: Can an active selection of training conditions speed up training for new communication settings?**
  - It is possible to wisely select the training conditions. By generating data that is more "surprising" with respect to what has already been observed, the training speeds up.
  - We show that when a meta-learning equalizer has access to characterize the conditions of how the next data is generated, it can lead to a reduction in the total number of frames needed to achieve a target performance value.
  - This translates into fewer channel simulation calls, and reduced training complexity.

- **Research Question 3: Is it possible to provide formal guarantees of reliability for learning-based communication protocols?**
  - Formal guarantees can be provided, once the scope changes from probabilistic prediction into predictive sets, with the set size reflecting the self-confidence of the prediction.
  - We apply conformal prediction for several communication systems problems, and validate the prediction reliability, specifically that the true label is indeed part of the predictive set in probability of no less than a target coverage rate.
  - We calibrate communication systems such as symbol demodulation and modulation classification for both VB-CP and CV-CP, and compare the performance of the schemes. The maximal number of folds K for N data points, as is set in N-CV-CP for $K = N$, is indeed the most efficient one. K-CV-CP, for some fixed $K < N$, is observed to be smaller in size than VB-CP, striking a balance between efficiency and computational complexity.
  - We exemplify online CP via RSS channel prediction, and show it is more efficient than the naïve -quantile-based predictor.

  - As a use-case, we show that URLLC scheduling via CP results in both a guaranteed reliability within the latency constraint of the URLLC service, while freeing resources for the eMBB traffic.
  - Lastly, we provide mathematical proof that CV-CRC, which is the outcome of the integration of CV-CP and VB-CRC, does indeed meet risk guarantees, and might lead to more efficient prediction sets than the state of the art CRC.

7.2 Open Research Questions

In this research, we have covered tools and applications for reliable learning-based communication systems. We investigated several communication problems like symbol demodulation, equalization, modulation classification, RSS prediction, and URLLC dynamic scheduling. Apart from the direct extensions for other communication systems and services, there are many other aspects, both in a machine learning prospective and in a communication system prospective, that can be the subject of future work. They will help to understand better how one can try and to have a reliable predictor, when faced with small amount of pilots, that serve as labeled data. At the same time, it is desired to do so while considering the predictors to be of good quality, for example high-accuracy or well-coverage. We detail questions that are still open, grouped into research fields.

- **Bayesian meta-learning:** Future work expanding Chapter 3 may consider a fully Bayesian meta-learning implementation that also accounts for uncertainty at the level of hyperparameters (see, e.g., [156] and references therein). This may be particularly useful in the regime of low number of frames. A study on the impact of well-calibrated decisions obtained via Bayesian learning on downstream blocks at the receiver, such as channel decoding, is also of interest. Moreover, the proposed tools of Bayesian meta-learning may find applications to other problems in communications, such as power control [54] and channel coding [57, 225].

- **Bayesian active meta-learning:** One possible extension of active learning discussed in Chapter 3, is the integration of both active learning and active meta-learning, forming Bayesian active meta-active learning. This will target reduction in the number of learning tasks, while reducing the number of samples within each learning task, with early work reported in [226]. Another direction for research would be to investigate different scoring functions for active meta-learning (see, e.g., [67]).

- **Conformal prediction:** A future research for this exciting field used in Chapter 4 can try to reduce the $1-2\alpha$ miscoverage guarantee of CV-CP into $1-\alpha$, as conjectured by [90] to be some kind of artifact of the proof. The applications of CP to other use cases in wireless communication systems, as well as extension of involving training-based calibration [80, 77] and/or meta-learning [117] is also of interest.

- **Conformal risk control** For the CV-CRC problem discussed in Chapter 6, one can try to use the jackknife+ of [90] instead of the jackknife-minmax used as we use in this work, possibly resulting in more efficient sets. Incorporating a cross-validation approach into time series CP [207] can further be explored. Relaxation of the exchangeability assumption for CRC as in [227] can be integrated with cross-validation techniques as well. Another possible direction of research can be integration of the softening mechanism as in meta-learning CV-CP [116] with the framework of an arbitrary risk as in CRC. This will result in meta-learning CV-CRC, aiming for small predictive set of arbitrary risk, when the per-task availability of data is limited.

- **Explainable AI:** In recent years, explainable AI gained interest, due to the desire of decision-makers to rely on AI predictions in a way that "makes sense" to a human, or provides information on how the prediction was made. For example, as reviewed in [89] and studied from an information-theoretic perspective in [228], the need to assess the influence of each input feature to the prediction output is one way to address explainability. The residual perturbation of the output, provided input features perturbation, suffers greatly from the predictor which is usually poorly-calibrated. Incorporating CP to explainable AI so the output is a set of labels can reflect the uncertainty via the set size. Several researchers have begun to explore some integration of CP and explainable AI [229], and this can be further elaborated as mentioned above.

Appendix A

Experiments Details for Chapter 3

Table A.1 summarizes the parameters used for the numerical experiments in Sec. 3.4 for demodulation and equalization. Throughout the simulations, we used `PyTorch` [230] adopting `autograd`'s option `create_graph` = `True` to allow the computational graph to calculate second-order derivatives.

For the demodulation problem in Sec. 3.4.2 (Figs. 3.6 – 3.8), the complex input space $\mathcal{X} = \mathbb{C}$ is treated as a two-dimensional real vector space $\mathbb{R}^2$ when is fed into the neural network demodulator. The KL term in (3.18) is suppressed by a multiplicative coefficient of 0.1, as a means to emphasize the average log-likelihood term should have over the prior. This is an approach known as generalized Bayesian inference [231, 125]. To handle the discrepancy in the number of pilots for adaptation during meta-training and meta-testing, i.e., $N_*^{\mathrm{tr}} > N_\tau^{\mathrm{tr}}$, we consider the following strategy akin to burn-in phase [39] during meta-testing as done in [49]: *(i)* start with I updates using learning rate η utilizing N_τ^{tr} pilots among the available N_*^{tr} pilots; *(ii)* then, additional $I_* - I$ updates are performed with reduced learning rate (5% of the original learning rate) with all available N_*^{tr} pilots. This strategy becomes particularly useful in practical scalable systems in which the number of pilots may change depending on the deployment environments.

As for the equalization setting in Sec. 3.4.3 (Figs. 3.9 – 3.10), we observe that reinitializing the hyperparameter ξ to a random value at each data acquisition iteration benefits meta-training in practice. While using the previous iteration's optimized hyperparameter vector ξ as the starting point for the current iteration is useful in reducing the computational complexity [232, 49], we found it beneficial not to do so in our equalization problem to avoid meta-overfitting especially in the few-frames (e.g., 10 frames) regime of interest.

Table A.1 Parameters for the demodulation and equalization meta-learning.

Description	Symbol	Demodulation (Sec. 3.4.2)	Equalization (Sec. 3.4.3)
Signal to noise ratio [dB]	SNR	18	6
Modulation	-	16-QAM	4-PAM
Neural network input dimension	integer	$\dim(\mathbb{C}) = 2$	$\dim(\mathbb{R}^2) = 2$
Neural network layers size [{hidden},output]	neurons per layer	$[10, 30, 30, 16]$	$[, 1]$
Activation of hidden layers	-	ReLU	NA
Activation of last layer	-	softmax	no activation
Meta-training frames mini-batch size	B	16	full batch (t)
Frame-specific learning rate	η	10^{-1}	$2 \cdot 10^{-3}$
Meta-learning rate	κ	10^{-3}	$5 \cdot 10^{-2}$
Number of pilots for frame-specific updates while meta-training	N_τ^{tr}	4	4
Number of pilots for meta-updates while meta-training	N_τ^{te}	3000	4
Number of pilots for frame-specific updates while meta-testing	N_*^{tr}	8	4
Number of symbols with each channel states while meta-testing	N_*^{te}	4000	1000
Number of inner SGD updates while meta-training	I	2	2
Number of inner SGD updates while meta-testing	I_*	200	2
Number of meta-updates while meta-training	I_{meta}	200	no. of tasks t
Ensemble size (for Bayesian framework only)	R	100	100
Assumed precision of equalizer	β	-	150
Number of frames forming the initial experience	t_{init}	-	3
Number of meta-training iterations	-	200	100
Number of meta-testing frames averaged over	-	50	100

Appendix B

Supplementary Material for Chapter 4

B.1 Full-Conformal Prediction

Unlike validation-based and cross-validation-based set predictions, full-conformal (FC) prediction [90] needs retraining of a model using the augmented data set $\mathcal{D} \cup \{(x, y')\}$ for every possible candidate pair (x, y'). More precisely, given a test input x, an FC set predictor is constructed by choosing all candidate labels y' that has a lower NC scores than a portion of (at least) α of the N examples, using the model that has been trained with the corresponding augmented data set $\mathcal{D} \cup \{(x, y')\}$. Mathematically, this yields the following *inclusive full-conformal (IFC)* set predictor

$$\Gamma^{\text{IFC}}(x|\mathcal{D}) = \Big\{ y' \in \mathcal{Y} \Big| \begin{array}{l} \text{NC}((x, y')|\mathcal{D} \cup \{(x, y')\}) \\ \qquad \leq Q_\alpha\Big\{\text{NC}(z[i]|\mathcal{D} \cup \{(x, y')\})\Big\}_{i=1}^{N} \Big\}. \end{array} \tag{B.1}$$

The term "inclusive" emphasizes that the points for which the NC scores apply ((x, y') and $z[i]$ for $i = 1, 2, \ldots, N$) are inclusive to the fitting sets. It mandates training $N^{\text{te}}|\mathcal{Y}|$ models, and predicting $N^{\text{te}}|\mathcal{Y}|(N + 1)$ times, which requires far more computations as compared to VB CP and CV CP predictors introduced in Sec. 4.4. By [161, Theorem 1], inclusion-based full-conformal predictor (B.1) is valid (2.32).

Since inclusive full-conformal (B.2) may suffer from overfitting due to inclusive nature, alternatively, the full-conformal approach can also use a *leave-one-out* data set from the augmented set $\mathcal{D} \cup \{(x, y')\}$ [17, 201, 18], meaning that the NC scores associated with

point $z[i]$ excludes itself during training, i.e., $\mathcal{D} \cup \{(x, y')\} \setminus \{z[i]\}$, whereas NC score of the prospective pairs $\{(x, y')\}$ uses plainly $\mathcal{D}$ for training (the augmented leaving the prospective pair out). We term this as *exclusive full-conformal (EFC)*, which is formally given as

$$\Gamma^{\mathrm{EFC}}(x|\mathcal{D}) = \Big\{ y' \in \mathcal{Y} \Big| \\ \mathrm{NC}((x, y')|\mathcal{D}) \leq Q_\alpha\Big\{\mathrm{NC}(z[i]|\mathcal{D} \cup \{(x, y')\} \setminus \{z[i]\})\Big\}_{i=1}^{N}\Big\}. \tag{B.2}$$

Obtaining (B.2) is highly computational complex, since the fitting algorithm needs to rerun $N+1$ times for each prospective pair, summing to a total of $N^{\mathrm{te}}|\mathcal{Y}|(N+1)$ training phases, which may be impractical to even moderate size problems. The number of model predictions is $N^{\mathrm{te}}|\mathcal{Y}|(N+1)$, same with IFC (B.1), yet this is far less computational complex than fitting. On the upside, the exclusive FC (B.2) is less prone to overfitting than the (inclusive) FC (B.1), due to its out of set nature. By [78, Theorem 1], exclusion-based full-conformal predictor (B.2) is valid (2.32).

B.2 Algorithmic Details for Rolling Conformal Inference

The RCI algorithm is reproduced from [233] in Algorithm 6.

B.3 Langevin Monte Carlo Approximation

One MC method to approximate the Bayesian predictor is to use the stochastic gradient Langevin dynamics (SGLD) [39], which is a first order scheme, hence less complex than other approaches, e.g., Hamiltonian Monte Carlo [41]. As we deal with exchangeable distributions, we focus on the full batch variant of it, which is known as Langevin Monte Carlo (LMC). It is an ensemble MC approximation as in (2.27), with the R model parameters being the consecutive outcomes of a noise injected gradient descent update steps for the optimization of the regularized log-loss that takes into account for the prior $p(\phi)$ as follows

$$\bar{\phi}_{r+1} \leftarrow \bar{\phi}_r - \eta\Big(\nabla_\phi L_{\mathcal{D}}(\bar{\phi}_r) - \tfrac{1}{N}\nabla_\phi \log p(\bar{\phi}_r)\Big) + \sqrt{\tfrac{2\eta}{NT}} v_r. \tag{B.3}$$

In the SGLD update of (B.3), $\bar{\phi}_r$ stands for the model parameter vector; v_r is a realization of the Gaussian random vector $\mathbf{v}_r \sim \mathcal{N}(0, I)$; η is the learning rate; and T is the temperature [140] that balances between the empirical loss and the injected noise power.

Algorithm 6: `Rolling Conformal Inference` (for Regression) [233]

Inputs : α = long-term target miscoverage level
$\theta[1]$ = initial calibration parameter
$\phi^{\text{lo}}[1], \phi^{\text{hi}}[1]$ = initial models

Parameters : I = number of online iterations
γ = learning rate for calibration parameter
η = learning rate for model updates

Output : $\{\Gamma_i^{\text{RCI}}(x[i]|\{z[j]\}_{j=1}^{i-1})\}_{i=1}^{I}$ = predicted sets for $\{x[i]\}_{i=1}^{I}$

for $i = 1, \ldots, I$ *time instants* **do**
  Retrieve a new data sample $(x[i], y[i])$
  ◁ *Set prediction of new input* ▷
  Calculate set $\Gamma_i^{\text{RCI}}(x[i]|\{z[j]\}_{j=1}^{i-1})$ using (4.19)
  $\Big[\hat{y}(x[i]|\phi^{\text{lo}}[i]) - \varphi(\theta[i]), \hat{y}(x[i]|\phi^{\text{hi}}[i]) + \varphi(\theta[i])\Big]$
  ◁ *Check if prediction is unsuccessful* ▷
  $\text{err}[i] \leftarrow \mathbb{1}(y[i] \notin \Gamma_i^{\text{RCI}}(x[i]|\{z[j]\}_{j=1}^{i-1}))$
  ◁ *Update calibration parameter* ▷
  $\theta[i+1] \leftarrow \theta[i] + \gamma(\text{err}[i] - \alpha)$
  ◁ *Update models using new sample* ▷
  $\phi^{\text{lo}}[i+1] \leftarrow \phi^{\text{lo}}[i] - \eta \nabla_\phi \ell_{\alpha/2}\Big(y[i], \hat{y}(x[i]|\phi^{\text{lo}}[i])\Big)$
  $\phi^{\text{hi}}[i+1] \leftarrow \phi^{\text{hi}}[i] - \eta \nabla_\phi \ell_{1-\alpha/2}\Big(y[i], \hat{y}(x[i]|\phi^{\text{hi}}[i])\Big)$
return predicted sets $\{\Gamma_i^{\text{RCI}}(x[i]|\{z[j]\}_{j=1}^{i-1})\}_{i=1}^{I}$

The initial model parameter $\bar{\phi}_1$ is set at random, and total of I gradient descent updates take place. The first updates are characterized by high absolute values of gradients, moving towards the maximum a posteriori (MAP) solution. As getting closer to it, the last noise injection term of (B.3) makes the model parameter vector to oscillate around it. It is a common practice to discard a burn-in period, the first $R_{\min}$ so that the ensemble is composed by the model parameter vectors of the last $R = I - R_{\min}$ iterations, using the ensemble $\{\phi_r = \bar{\phi}_{R_{\min}+r}\}_{r=1}^{R}$. This is a MC approximation, as the model parameter vectors forming the ensemble are considered to be drawn from the posterior $p(\phi|\mathcal{D})$. The noise injection in the update step (B.3) enables the exploration of the proximity around the minima, and the ensemble sampling allows to maintain some quantity of uncertainty around the frequentist solution.

B.4 Implementation of Online Channel Prediction

The architecture of the set predictor is inspired by [233], and made out of three artificial neural networks. The first, $f_{\text{pre}}(\cdot)$, is a multi-layer perceptron (MLP) network with hidden layers of $16, 32$ neurons each, parametrized by vector $\phi_{\text{pre}}[i]$. It is meant to apply a pre-process over the most recent observed $K = 20$ pairs $\{z[i-K], \ldots, z[i-1]\}$ to be transformed element-wise into a length-K vector $w[i] = \Big[w_1[i], \ldots, w_K[i]\Big]^\top$, in which the k-th element $(k = 1, \ldots, K)$ is $w_k[i] = f_{\text{pre}}\Big(z[i-K+k-1]\Big|\phi_{\text{pre}}[i]\Big)$. Effectively, this will serve as a temporal sliding K-length window, with a time-evolving pre-processing function. The second neural network, $f_{\text{LSTM}}(\cdot)$ has two layers with model parameter vectors $\phi^1_{\text{LSTM}}[i]$ (first layer) and $\phi^2_{\text{LSTM}}[i]$ (second layer), which retains a memory via the hidden state vectors h and c, initialized at every time index i as $c^1_0[i] = c^2_0[i] = h^1_0[i] = h^2_0[i] = \mathbf{0}$. By accessing the previous K pairs via the vector $w[i]$, this recurrent neural network extracts temporal patterns by sequentially transferring information via LSTM cells (with shared parameter vectors) in the image of hidden and cell state vectors $c_k[i], h_k[i]$ via the LSTM cells. These vectors flow along the LSTM by concatenating $k = 1, \ldots, K$ cells, and forming vectors of length 32 each

$$[c^1_k[i], h^1_k[i]] = f_{\text{LSTM}}\Big(w_k[i], c^1_{k-1}[i], h^1_{k-1}[i]\Big|\phi^1_{\text{LSTM}}[i]\Big) \tag{B.4}$$

$$[c^2_k[i], h^2_k[i]] = f_{\text{LSTM}}\Big(h^1_k[i], c^2_{k-1}[i], h^2_{k-1}[i]\Big|\phi^2_{\text{LSTM}}[i]\Big). \tag{B.5}$$

The third and last network is a post-processing MLP $f_{\text{post}}(\cdot)$ with one hidden layer of 32 neurons, and with parameter vector $\phi_{\text{post}}[i]$, which maps the last LSTM hidden 64-length vector $h_K[i] = [h^1_K[i], h^2_K[i]]^\top$ into a scalar $f_{\text{post}}\Big(x[i], h_K[i]\Big|\phi_{\text{post}}[i]\Big) \in \mathbb{R}$ that estimates the quantile for the output $y[i]$. Accordingly, the time evolving model parameter is the tuple

$$\phi[i] = (\phi_{\text{pre}}[i], \phi^1_{\text{LSTM}}[i], \phi^2_{\text{LSTM}}[i], \phi_{\text{post}}[i]). \tag{B.6}$$

This model is instantiated twice for the regression problem: one for the $\alpha/2$ lower quantile and the other for $1-\alpha/2$ upper quantile. For every time instant i, after the new output $y[i]$ is observed, continual learning of the models is taken place by training the models with corresponding pinball losses (4.5) using the new pair $(x[i], y[i])$, while initializing the models as the previous models at time instant $i-1$. Full design details are in Table B.1.

The miscoverage rate was set to $\alpha = 0.1$, the learning rate to $\eta = 0.01$, and we chose $\gamma = 0.03$ for the calibration parameter θ in (4.18).

parameter	symbol	Zanella [1]	Simmons [2]
sequence length	K	20	20
input size	$\dim(x[i])$	1	0
f_{pre} input size	$\dim(z[i-k])$	2	1
f_{pre} hidden layers	-	$[16, 32]$	$[16, 32]$
f_{LSTM} hidden size	$\dim(h_k[i]) = \dim(c_k[i])$	32	32
f_{LSTM} number of layers	-	2	2
f_{post} input size	$\dim(h_K[i])$	64	64
f_{post} hidden layers	-	[32]	[32]

Table B.1 RSS prediction neural networks hyperparameters

Appendix C

Supplementary Material for Chapter 6

C.1 Proof that VB-CRC achieves target risk

In this appendix, we prove condition (6.13) for VB-CRC. While this result was originally shown in [91], here we provide an equivelant proof that is more convenient to support the proof of Theorem 1 in Appendix C.3. We start by bounding the VB-CRC threshold (6.11) using the following steps

$$\begin{aligned}
\lambda^{\mathrm{VB}}(\mathcal{D}^{\mathrm{val}}|\mathcal{D}^{\mathrm{tr}}) &= \inf_{\lambda}\left\{\lambda\middle|\tfrac{1}{N^{\mathrm{val}}+1}\left(\sum_{i=1}^{N^{\mathrm{val}}}\ell\Big(y^{\mathrm{val}}[i],\Gamma_{\lambda}(x^{\mathrm{val}}[i]|\mathcal{D}^{\mathrm{tr}})\Big)+B\right)\leq\alpha\right\}\\
&\geq \inf_{\lambda}\left\{\lambda\middle|\tfrac{1}{N^{\mathrm{val}}+1}\left(\sum_{i=1}^{N^{\mathrm{val}}}\ell\Big(y^{\mathrm{val}}[i],\Gamma_{\lambda}(x^{\mathrm{val}}[i]|\mathcal{D}^{\mathrm{tr}})\Big)+\ell\Big(y,\Gamma_{\lambda}(x|\mathcal{D}^{\mathrm{tr}})\Big)\right)\leq\alpha\right\}\\
&=: \lambda'(\mathcal{D}^{\mathrm{val}},x,y|\mathcal{D}^{\mathrm{tr}}), \qquad \text{(C.1)}
\end{aligned}$$

where the inequality in (C.1) follows from (6.4). The ground-truth risk averaged over test example (x,y) and validation set $\mathcal{D}^{\mathrm{val}}$ is upper bounded as

$$\begin{aligned}
&\mathrm{E}_{\boldsymbol{\mathcal{D}}^{\mathrm{val}},\mathbf{x},\mathbf{y}\sim p_0(\mathcal{D}^{\mathrm{val}},x,y)}\left[\ell\Big(\mathbf{y},\Gamma_{\lambda^{\mathrm{VB}}(\boldsymbol{\mathcal{D}}^{\mathrm{val}}|\mathcal{D}^{\mathrm{tr}})}(\mathbf{x}|\mathcal{D}^{\mathrm{tr}})\Big)\right]\\
&\qquad\leq \mathrm{E}_{\boldsymbol{\mathcal{D}}^{\mathrm{val}},\mathbf{x},\mathbf{y}\sim p_0(\mathcal{D}^{\mathrm{val}},x,y)}\left[\ell\Big(\mathbf{y},\Gamma_{\lambda'(\boldsymbol{\mathcal{D}}^{\mathrm{val}},\mathbf{x},\mathbf{y}|\mathcal{D}^{\mathrm{tr}})}(\mathbf{x}|\mathcal{D}^{\mathrm{tr}})\Big)\right] \qquad \text{(C.2a)}\\
&\qquad\leq \alpha, \qquad \text{(C.2b)}
\end{aligned}$$

where the first inequality (C.2a) follows the nesting property (6.8) given inequality (C.1). The second inequality (C.2b) is an application of the following lemma, whose proof is deferred to Appendix C.2.

Lemma 2. *Let* $\mathbf{v}_1, \ldots, \mathbf{v}_M$ *be random variables with an exchangeable joint distribution such that the equation* $\mathbb{P}\Big(\frac{1}{M}\sum_{i=1}^{M}\mathbf{v}_i \leq \alpha\Big) = 1$ *holds. Then, we have the inequality* $\mathrm{E}_{\mathbf{v}_{1:M}\sim p_0(v_{1:M})}[\mathbf{v}_m] \leq \alpha$ *for all* $m \in \{1, ..., M\}$.

To apply Lemma 2 in (C.2b), we define $M = N^{\mathrm{val}} + 1$ variables by

$$\mathbf{v}_i = \begin{cases} \ell\Big(\mathbf{y}^{\mathrm{val}}[i], \Gamma_{\lambda'(\boldsymbol{\mathcal{D}}^{\mathrm{val}}, \mathbf{x}, \mathbf{y}|\mathcal{D}^{\mathrm{tr}})}(\mathbf{x}^{\mathrm{val}}[i]|\mathcal{D}^{\mathrm{tr}})\Big) & i = 1, \ldots, N^{\mathrm{val}} \\ \ell\Big(\mathbf{y}, \Gamma_{\lambda'(\boldsymbol{\mathcal{D}}^{\mathrm{val}}, \mathbf{x}, \mathbf{y}|\mathcal{D}^{\mathrm{tr}})}(\mathbf{x}|\mathcal{D}^{\mathrm{tr}}) & i = N^{\mathrm{val}} + 1, \end{cases} \tag{C.3}$$

whose empirical average is, by (C.1), no greater than α. Furthermore, to comply with the technical conditions of Lemma 2, variables $\mathbf{v}_{1:M}$ need to be exchangeable. This is justified by the following lemma, which is a corollary of [234, Theorem 3] or [235, Theorem 4].

Lemma 3. *Let* $\mathbf{w}_1, \ldots, \mathbf{w}_M \in \mathcal{W}$ *be a collection of exchangeable random vectors,* $f : \mathcal{W} \to \mathbb{R}$ *be a fixed mapping, and* $g : \mathcal{W}^M \to \mathbb{R}$ *be a fixed mapping that is permutation-invariant, i.e., oblivious to the ordering of its* M *input values. Then, the* M *random variables formed as* $\mathbf{v}_1 = f(\mathbf{w}_1, g(\mathbf{w}_{1:M}))$, $\ldots$, $\mathbf{v}_M = f(\mathbf{w}_M, g(\mathbf{w}_{1:M}))$ *are exchangeable.*

Lemma 3 implies the exchangeability of variables (C.3) by defining the $N^{\mathrm{val}} + 1$ exchangeable vectors as

$$\mathbf{w}_i = \begin{cases} \mathbf{z}^{\mathrm{val}}[i] & i = 1, \ldots, N^{\mathrm{val}} \\ (\mathbf{x}, \mathbf{y}) & i = N^{\mathrm{val}} + 1; \end{cases} \tag{C.4}$$

the permutation invariant function is set as $g(\cdot) = \lambda'(\cdot|\mathcal{D}^{\mathrm{tr}})$; the fixed mapping is

$$\mathbf{v}_i = f\Big(\mathbf{w}_i = (\mathbf{x}_i, \mathbf{y}_i), g(\mathbf{w}_{1:M})\Big) = \ell(\mathbf{y}_i, \Gamma_{g(\mathbf{w}_{1:M})}(\mathbf{x}_i|\mathcal{D}^{\mathrm{tr}})); \tag{C.5}$$

and we focus on the average risk of the last term, i.e., $m = M = N^{\mathrm{val}} + 1$. This completes the proof of (6.13).

C.2 Proof of Lemma 2

In this appendix, we prove Lemma 2. To start, define a *bag* $u = \wr u_1, \ldots, u_M \int$ of M elements $u_1 \ldots, u_M$ as a multiset, i.e., as an unordered list with allowed repetitions [17]. By definition,

two bags u and v are equal if they contain the same elements, irrespective of the ordering of their identical items, which we write as $u \overset{\text{bag}}{=} v$. One can form a bag out of a random vector $\mathbf{v}_1, \ldots, \mathbf{v}_M \sim p_0(v_{1:M})$ by discarding the order of the items. Accordingly, the distribution of the bag $\mathbf{u}$ is given by

$$p_0(u) = \mathbb{P}\Big(\{\!\{\mathbf{v}_1, ..., \mathbf{v}_M\}\!\} \overset{\text{bag}}{=} \{\!\{u_1, \ldots, u_M\}\!\}\Big) = \sum_{\pi \in \Pi_M} \mathbb{P}\Big(\mathbf{v}_1 = u_{\pi(1)}, ..., \mathbf{v}_M = u_{\pi(M)}\Big), \quad \text{(C.6)}$$

where the sum is over the set Π_M of all $M!$ permutations. For example, three Bernoulli variables $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 \underset{\text{i.i.d.}}{\sim} \text{Bern}(q)$ with parameter $q \in [0, 1]$ can constitute four different bags. In fact, bag $\mathbf{u} \overset{\text{bag}}{=} \{\!\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}\!\}$ equals $\{\!\{0, 0, 0\}\!\}$ with probability (w.p.) $(1-q)^3$, $\{\!\{0, 0, 1\}\!\}$ w.p. $3(1-q)^2 q$, $\{\!\{0, 1, 1\}\!\}$ w.p. $3(1-q)q^2$, and $\{\!\{1, 1, 1\}\!\}$ w.p. q^3.

With these definitions, we obtain the following chain of inequalities

$$\begin{aligned}
\mathrm{E}_{\mathbf{v}_{1:M} \sim p_0(v_{1:M})}[\mathbf{v}_m] &= \mathrm{E}_{\mathbf{u} \sim p_0(u)}\Big[\mathrm{E}_{\mathbf{v}_{1:M} \sim p_0(v_{1:M}|u)}\Big[\mathbf{v}_m \Big| \{\!\{\mathbf{v}_1, ..., \mathbf{v}_M\}\!\} \overset{\text{bag}}{=} \mathbf{u}\Big]\Big] && \text{(C.7a)} \\
&= \mathrm{E}_{\mathbf{u} \sim p_0(u)}\Big[\tfrac{1}{M}\sum_{l=1}^{M} \mathbf{u}_l\Big] && \text{(C.7b)} \\
&= \mathrm{E}_{\mathbf{v}_{1:M} \sim p_0(v_{1:M})}\Big[\mathrm{E}_{\mathbf{u} \sim p_0(u|v_{1:M})}\Big[\tfrac{1}{M}\sum_{l=1}^{M} \mathbf{u}_l \Big| \mathbf{u} \overset{\text{bag}}{=} \{\!\{\mathbf{v}_1, \ldots, \mathbf{v}_M\}\!\}\Big]\Big] && \text{(C.7c)} \\
&= \mathrm{E}_{\mathbf{v}_{1:M} \sim p_0(v_{1:M})}\Big[\mathrm{E}_{\mathbf{u} \sim p_0(u|v_{1:M})}\Big[\tfrac{1}{M}\sum_{l=1}^{M} \mathbf{v}_l \Big| \mathbf{u} \overset{\text{bag}}{=} \{\!\{\mathbf{v}_1, \ldots, \mathbf{v}_M\}\!\}\Big]\Big] && \text{(C.7d)} \\
&= \mathrm{E}_{\mathbf{v}_{1:M} \sim p_0(v_{1:M})}\Big[\tfrac{1}{M}\sum_{l=1}^{M} \mathbf{v}_l\Big] && \text{(C.7e)} \\
&\leq \alpha. && \text{(C.7f)}
\end{aligned}$$

The inequalities of (C.7) are justified as follows: (C.7a) and (C.7c) stem from the law of iterated expectations over all possible bags of M items; (C.7b) arises from the fact that each item in the bag has an equal likelihood to be the realization of the m-th variable $\mathbf{v}_m$; is again the law of iterated expectation with the reintroduction of the random vector; (C.7d) stems from the fact that if two bags have the same items, their sum is identical; (C.7e) leverages the fact that the bag given its random variables is a deterministically specified; and lastly, (C.7f) is by the assumption in Lemma 2. This concludes the proof of Lemma 2.

C.3 Proof of Theorem 1

To prove Theorem 1, let us introduced an augmented data set, such that the last, $(K+1)$-th, fold $\mathcal{D}^{\text{te}} = \mathcal{D}_{K+1}$ is composed of N/K arbitrary test points

$$\tilde{\mathcal{D}} = \Big\{ \underbrace{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K}_{=\mathcal{D}}, \underbrace{\mathcal{D}_{K+1}}_{=\mathcal{D}^{\text{te}}} \Big\} \tag{C.8}$$

with the test point (x, y) included as the first point in the test set, i.e., $(x, y) = (x^{\text{te}}[1], y^{\text{te}}[1]) = (x_{K+1}[1], y_{K+1}[1])$. By construction, all $N + N/K$ points in the augmented data set $\tilde{\mathcal{D}}$ are exchangeable and distributed according to joint distribution $p_0(\tilde{\mathcal{D}}) = p_0(\mathcal{D}, \mathcal{D}^{\text{te}})$. We denote the elements of the augmented set $\tilde{\mathcal{D}}$ in (C.8) as

$$(\tilde{x}_k[j], \tilde{y}_k[j]) = (x_k[j], y_k[j]) \text{ for } k \in \{1, \ldots K\} \tag{C.9a}$$

$$(\tilde{x}_{K+1}[j], \tilde{y}_{K+1}[j]) = (x^{\text{te}}[j], y^{\text{te}}[j]). \tag{C.9b}$$

Note that the augmented set $\tilde{\mathcal{D}}$ in (C.8) is different than the augmented set using dummy points $\mathcal{D}^{\text{aug}}$ (6.17). For a pair of folds indices $k', k \in \{1, \ldots, K+1\}$ with $k \neq k'$, we also define the augmented leave-two-folds-out (L2O) set as the augmented set without the two indexed folds, i.e.,

$$\tilde{\mathcal{D}}_{-(k',k)} = \tilde{\mathcal{D}} \setminus \{\mathcal{D}_{k'}, \mathcal{D}_k\}. \tag{C.10}$$

As a special case, when one of the indices points to the $(K+1)$-th fold, which is the test fold, the L2O reduces to the leave-one-out of the available data set $\tilde{\mathcal{D}}_{-(K+1,k)} = \mathcal{D}_{-k}$. For every fold within the augmented data set $\tilde{\mathcal{D}}$ (C.8), we evaluate the average L2O loss (C.10), minimized over the second fold index as

$$\hat{R}^{\text{CV}}_{\text{L2O}}(\lambda|\tilde{\mathcal{D}}) = \tfrac{1}{K+1} \sum_{k=1}^{K+1} \tfrac{K}{N} \sum_{j=1}^{N/K} \min_{k' \in \{1,\ldots,K+1\} \setminus \{k\}} \Big\{ \ell\Big(\tilde{y}_k[j], \Gamma_\lambda(\tilde{x}_k[j]|\tilde{\mathcal{D}}_{-(k,k')})\Big) \Big\}. \tag{C.11}$$

Finally, we define the L2O threshold as the minimal threshold value for which the estimated average L2O risk (C.11) is no larger than α, i.e.,

$$\lambda^{\text{CV}}_{\text{L2O}}(\tilde{\mathcal{D}}) = \inf_{\lambda} \Big\{ \lambda \Big| \hat{R}^{\text{CV}}_{\text{L2O}}(\lambda|\tilde{\mathcal{D}}) \leq \alpha \Big\}. \tag{C.12}$$

Corollary 4. *The L2O threshold* $\lambda^{CV}_{L2O}(\tilde{\mathcal{D}})$ *in* (C.12) *is fold-permutation-invariant, i.e., for any of the* $(K+1)!$ *possible fold-permutation mappings* π*, we have*

$$\lambda^{CV}_{L2O}\Big(\tilde{\mathcal{D}}\Big) = \lambda^{CV}_{L2O}\Big(\{\tilde{\mathcal{D}}_k\}_{k=1}^{K+1}\Big) = \lambda^{CV}_{L2O}\Big(\{\tilde{\mathcal{D}}_{\pi[k]}\}_{k=1}^{K+1}\Big). \tag{C.13}$$

This is due to the commutative property of the outer fold-summation and of the inner, within-fold, summation in (C.11).

Lemma 5. *The L2O threshold* $\lambda^{CV}_{L2O}(\tilde{\mathcal{D}})$ *in* (C.12) *lower bounds the K-CV-CRC threshold* (6.18)

$$\lambda^{CV}_{L2O}(\tilde{\mathcal{D}}) \leq \lambda^{CV}(\mathcal{D}). \tag{C.14}$$

The proof of Lemma 5 is given in Appendix C.4.

We now define $K+1$ random variables $\mathbf{v}_1, \ldots, \mathbf{v}_{K+1}$, whose randomness stems from their dependence on the augmented data set $\tilde{\boldsymbol{\mathcal{D}}}$. Each k-th random variable $\mathbf{v}_k$ is the minimal leave-two-fold-out empirical risk averaged over the N/K examples in the validation fold $\boldsymbol{\mathcal{D}}_k$, i.e.,

$$\mathbf{v}_k = \tfrac{K}{N} \sum_{j=1}^{N/K} \min_{k' \in \{1,\ldots,K+1\}\setminus\{k\}} \Big\{ \ell\Big(\tilde{\mathbf{y}}_k[j], \Gamma_{\lambda^{\mathrm{CV}}_{\mathrm{L2O}}(\tilde{\boldsymbol{\mathcal{D}}})}(\tilde{\mathbf{x}}_k[j] | \tilde{\boldsymbol{\mathcal{D}}}_{-(k',k)})\Big)\Big\} \quad \text{for} \quad k = 1, \ldots, K+1. \tag{C.15}$$

The random variables $\Big\{\mathbf{v}_1, \ldots, \mathbf{v}_{K+1}\Big\} = \Big\{v_1(\lambda, \tilde{\boldsymbol{\mathcal{D}}}), \ldots, v_{K+1}(\lambda, \tilde{\boldsymbol{\mathcal{D}}})\Big\}$ are exchangeable for any fixed threshold due to the exchangeability of the folds in the augmented data set. Therefore, by Lemma 2, we have the inequality

$$\mathrm{E}_{\tilde{\boldsymbol{\mathcal{D}}} \sim p_0(\tilde{\mathcal{D}})}\Big[v_{K+1}(\lambda^{\mathrm{CV}}_{\mathrm{L2O}}(\tilde{\boldsymbol{\mathcal{D}}}), \tilde{\boldsymbol{\mathcal{D}}})\Big] \leq \alpha. \tag{C.16}$$

We are now ready to follow the steps

$$\begin{aligned}
&\mathrm{E}_{\boldsymbol{\mathcal{D}},\mathbf{x},\mathbf{y}\sim p_0(\mathcal{D},x,y)}\Big[\ \ell\Big(\ \mathbf{y}, \Gamma^{\mathrm{CV}}(\mathbf{x}|\boldsymbol{\mathcal{D}})\Big)\Big] \\
&\qquad = \mathrm{E}_{\boldsymbol{\mathcal{D}},\mathbf{x},\mathbf{y}\sim p_0(\mathcal{D},x,y)}\Big[\ell\Big(\mathbf{y}, \bigcup_{k'=1}^{K} \Gamma_{\lambda^{\mathrm{CV}}(\boldsymbol{\mathcal{D}})}(\mathbf{x}|\boldsymbol{\mathcal{D}}_{-k'})\Big)\Big] && \text{(C.17a)} \\
&\qquad \leq \mathrm{E}_{\boldsymbol{\mathcal{D}},\mathbf{x},\mathbf{y}\sim p_0(\mathcal{D},x,y)}\Big[\min_{k' \in \{1,\ldots,K\}} \Big\{\ell\Big(\mathbf{y}, \Gamma_{\lambda^{\mathrm{CV}}(\boldsymbol{\mathcal{D}})}(\mathbf{x}|\boldsymbol{\mathcal{D}}_{-k'})\Big)\Big\}\Big] && \text{(C.17b)} \\
&\qquad = \mathrm{E}_{\boldsymbol{\mathcal{D}},\boldsymbol{\mathcal{D}}^{\mathrm{te}}\sim p_0(\mathcal{D},\mathcal{D}^{\mathrm{te}})}\Big[\min_{k' \in \{1,\ldots,K\}} \Big\{\ell\Big(\mathbf{y}^{\mathrm{te}}[1], \Gamma_{\lambda^{\mathrm{CV}}(\boldsymbol{\mathcal{D}})}(\mathbf{x}^{\mathrm{te}}[1]|\boldsymbol{\mathcal{D}}_{-k'})\Big)\Big\}\Big] && \text{(C.17c)} \\
&\qquad = \mathrm{E}_{\boldsymbol{\mathcal{D}},\boldsymbol{\mathcal{D}}^{\mathrm{te}}\sim p_0(\mathcal{D},\mathcal{D}^{\mathrm{te}})}\Big[\tfrac{K}{N}\sum_{j=1}^{N/K} \min_{k' \in \{1,\ldots,K\}} \Big\{\ell\Big(\mathbf{y}^{\mathrm{te}}[j], \Gamma_{\lambda^{\mathrm{CV}}(\boldsymbol{\mathcal{D}})}(\mathbf{x}^{\mathrm{te}}[j]|\boldsymbol{\mathcal{D}}_{-k'})\Big)\Big\}
\end{aligned}$$

$$= \mathrm{E}_{\tilde{\boldsymbol{\mathcal{D}}}\sim p_0(\tilde{\mathcal{D}})}\Big[\min_{k'\in\{1,\dots,K\}}\Big\{\tfrac{K}{N}\sum_{j=1}^{N/K}\ell\Big(\tilde{\mathbf{y}}_{K+1}[j],\Gamma_{\lambda^{\mathrm{CV}}(\tilde{\boldsymbol{\mathcal{D}}})}(\tilde{\mathbf{x}}_{K+1}[j]|\tilde{\boldsymbol{\mathcal{D}}}_{-(k',K+1)})\Big)\Big\}\Big] \tag{C.17d}$$

$$\leq \mathrm{E}_{\tilde{\boldsymbol{\mathcal{D}}}\sim p_0(\tilde{\mathcal{D}})}\Big[\min_{k'\in\{1,\dots,K\}}\Big\{\tfrac{K}{N}\sum_{j=1}^{N/K}\ell\Big(\tilde{\mathbf{y}}_{K+1}[j],\Gamma_{\lambda^{\mathrm{CV}}_{\mathrm{L2O}}(\tilde{\boldsymbol{\mathcal{D}}})}(\tilde{\mathbf{x}}_{K+1}[j]|\tilde{\boldsymbol{\mathcal{D}}}_{-(k',K+1)})\Big)\Big\}\Big] \tag{C.17e}$$

$$\leq \alpha, \tag{C.17f}$$

$$\leq \alpha, \tag{C.17g}$$

where (C.17a) is a consequence of (6.15), which is equivalent to $\Gamma^{\mathrm{CV}}_{\lambda}(x|\mathcal{D}) = \bigcup_{k=1}^{K}\Gamma_{\lambda}(x|\mathcal{D}_{-k})$; inequality (C.17b) is due to the nesting property (6.5) applied on a particular left-fold-out k' which is a subset of the union of all left-fold-out sets; (C.17d) leverages exchangeability as all test points have the same expected loss; (C.17e) uses the augmented data set notations (C.9b); inequality (C.17f) is an outcome of the nesting properties (6.8) and (6.5) with inequality (C.14); in inequality (C.17g), we have used (C.16), alongside Corollary 4, stating that the L2O threshold is fold-invariant. This completes the proof of Theorem 1.

C.4 Proof of Lemma 5

The proof of Lemma 5 stated in Appendix C.3 follows the steps

$$\lambda^{\mathrm{CV}}_{\mathrm{L2O}}(\tilde{\mathcal{D}}) = \inf_{\lambda}\left\{\lambda\,\middle|\,\tfrac{1}{K+1}\sum_{k=1}^{K+1}\tfrac{K}{N}\sum_{j=1}^{N/K}\min_{k'\in\{1,\dots,K+1\}\setminus\{k\}}\Big\{\ell\Big(\tilde{y}_k[j],\Gamma_{\lambda}(\tilde{x}_k[j]|\tilde{\mathcal{D}}_{-(k,k')})\Big)\Big\}\leq\alpha\right\} \tag{C.18a}$$

$$= \inf_{\lambda}\left\{\lambda\,\middle|\,\tfrac{1}{K+1}\left(\sum_{k=1}^{K}\tfrac{K}{N}\sum_{j=1}^{N/K}\min_{k'\in\{1,\dots,K+1\}\setminus\{k\}}\Big\{\ell\Big(\tilde{y}_k[j],\Gamma_{\lambda}(\tilde{x}_k[j]|\tilde{\mathcal{D}}_{-(k,k')})\Big)\Big\}\right.\right. \tag{C.18b}$$

$$\left.\left.+\tfrac{K}{N}\sum_{j=1}^{N/K}\min_{k'\in\{1,\dots,K\}}\Big\{\ell\Big(\tilde{y}_{K+1}[j],\Gamma_{\lambda}(\tilde{x}_{K+1}[j]|\tilde{\mathcal{D}}_{-(K+1,k')})\Big)\Big\}\right)\leq\alpha\right\}$$

$$\leq \inf_{\lambda}\left\{\lambda\,\middle|\,\tfrac{1}{K+1}\Big(\sum_{k=1}^{K}\tfrac{K}{N}\sum_{j=1}^{N/K}\min_{k'\in\{1,\dots,K+1\}\setminus\{k\}}\Big\{\ell\Big(\tilde{y}_k[j],\Gamma_{\lambda}(\tilde{x}_k[j]|\tilde{\mathcal{D}}_{-(k,k')})\Big)\Big\}+\tfrac{K}{N}\sum_{j=1}^{N/K}B\Big)\leq\alpha\right\} \tag{C.18c}$$

$$\leq \inf_{\lambda}\left\{\lambda\,\middle|\,\tfrac{1}{K+1}\Big(\sum_{k=1}^{K}\tfrac{K}{N}\sum_{j=1}^{N/K}\ell\Big(\tilde{y}_k[j],\Gamma_{\lambda}(\tilde{x}_k[j]|\tilde{\mathcal{D}}_{-(k,K+1)})\Big)+B\Big)\leq\alpha\right\} \tag{C.18d}$$

$$= \inf_{\lambda}\left\{\lambda\,\middle|\,\tfrac{1}{K+1}\Big(\sum_{k=1}^{K}\tfrac{K}{N}\sum_{j=1}^{N/K}\ell\Big(y_k[j],\Gamma_{\lambda}(x_k[j]|\mathcal{D}_{-k})\Big)+B\Big)\leq\alpha\right\} \tag{C.18e}$$

$$= \lambda^{\mathrm{CV}}(\mathcal{D}), \tag{C.18f}$$

where (C.18a) stems from the definition in (C.12); (C.18b) is obtained by decomposing the first sum into its first K summation terms, and by listing the last term, the $(K+1)$-th, on its own; and (C.18f) follows the definition in (6.18). This completes the proof of Lemma 5.

References

[1] A. Zanella and A. Bardella, "RSS-based Ranging by Multichannel RSS Averaging," *IEEE Wireless Communications Letters*, vol. 3, no. 1, pp. 10–13, 2013.

[2] N. Simmons, S. B. F. Gomes, M. D. Yacoub, O. Simeone, S. L. Cotton, and D. E. Simmons, "AI-Based Channel Prediction in D2D Links: An Empirical Validation," *IEEE Access*, vol. 10, pp. 65 459–65 472, 2022.

[3] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y.-J. A. Zhang, "The roadmap to 6G: AI Empowered Wireless Networks," *IEEE Communications Magazine*, vol. 57, no. 8, pp. 84–90, 2019.

[4] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and learning in O-RAN for data-driven NextG cellular networks," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 21–27, 2021.

[5] P. H. Masur, J. H. Reed, and N. Tripathi, "Artificial Intelligence in Open-Radio Access Network," *IEEE Aerospace and Electronic Systems Magazine*, 2022.

[6] O.-R. Alliance, "O-RAN Working Group 2 AI/ML Workflow Description and Requirements," *ORAN-WG2. AIML. v01.02.02*, vol. 1, 2020.

[7] B. Wang and K. R. Liu, "Advances in Cognitive Radio Networks: A Survey," *IEEE Journal of selected topics in signal processing*, vol. 5, no. 1, pp. 5–23, 2010.

[8] O. Simeone, I. Stanojev, S. Savazzi, Y. Bar-Ness, U. Spagnolini, and R. Pickholtz, "Spectrum Leasing to Cooperating Secondary Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 1, pp. 203–213, 2008.

[9] O. Simeone, S. Park, and J. Kang, "From Learning to Meta-Learning: Reduced Training Overhead and Complexity for Communication Systems," in *Proc. 2020 2nd 6G Wireless Summit (6G SUMMIT) in Levi, Finland.* IEEE, 2020, pp. 1–5.

[10] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in *International Conference on Machine Learning.* PMLR, 2017, pp. 1321–1330.

[11] A. Masegosa, "Learning under Model Misspecification: Applications to Variational and Ensemble Methods," *Proc. Advances in Neural Information Processing Systems (NIPS) as Virtual-only Conference*, vol. 33, pp. 5479–5491, 2020.

[12] W. R. Morningstar, A. Alemi, and J. V. Dillon, "PACm-Bayes: Narrowing the Empirical Risk Gap in the Misspecified Bayesian Regime," in *Proc. International Conference on Artificial Intelligence and Statistics, as a Virtual-only Conference*. PMLR, 2022, pp. 8270–8298.

[13] M. Zecchin, S. Park, O. Simeone, M. Kountouris, and D. Gesbert, "Robust PACm: Training Ensemble Models Under Misspecification and Outliers," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[14] P. Cannon, D. Ward, and S. M. Schmon, "Investigating the Impact of Model Misspecification in Neural Simulation-based Inference," *arXiv preprint arXiv:2209.01845*, 2022.

[15] D. T. Frazier, C. P. Robert, and J. Rousseau, "Model Misspecification in Approximate Bayesian Computation: Consequences and Diagnostics," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 82, no. 2, pp. 421–444, 2020.

[16] J. Ridgway, "Probably Approximate Bayesian Computation: Nonasymptotic Convergence of ABC under Misspecification," *arXiv preprint arXiv:1707.05987*, 2017.

[17] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic Learning in a Random World*. Springer, 2005, springer, New York.

[18] M. Fontana, G. Zeni, and S. Vantini, "Conformal Prediction: A Unified Review of Theory and New Challenges," *Bernoulli*, vol. 29, no. 1, pp. 1 – 23, 2023.

[19] 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.

[20] O. Simeone, "A Very Brief Introduction to Machine Learning with Applications to Communication Systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 4, pp. 648–664, 2018.

[21] L. Dai, R. Jiao, F. Adachi, H. V. Poor, and L. Hanzo, "Deep Learning for Wireless Communications: An Emerging Interdisciplinary Paradigm," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 133–139, 2020.

[22] Y. Eldar, A. Goldsmith, D. Gündüz, and H. Poor, *Machine Learning and Wireless Communications*. Cambridge University Press, 2022.

[23] T. Erpek, T. J. O'Shea, Y. E. Sagduyu, Y. Shi, and T. C. Clancy, *Deep Learning for Wireless Communications*. Cham: Springer International Publishing, 2020, pp. 223–266.

[24] Z. Sun, J. Wu, X. Li, W. Yang, and J.-H. Xue, "Amortized Bayesian Prototype Meta-learning: A New Probabilistic Meta-learning Approach to Few-shot Image Classification," in *Proc. International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 1414–1422.

[25] K. M. Cohen, S. Park, O. Simeone, and S. Shamai, "Bayesian Active Meta-Learning for Reliable and Efficient AI-Based Demodulation," *IEEE Transactions on Signal Processing*, vol. 70, pp. 5366–5380, 2022.

[26] M. Zecchin, S. Park, O. Simeone, M. Kountouris, and D. Gesbert, "Robust Bayesian Learning for Reliable Wireless AI: Framework and Applications," *IEEE Transactions on Cognitive Communications and Networking*, 2023.

[27] E. Angelino, M. J. Johnson, R. P. Adams *et al.*, "Patterns of Scalable Bayesian Inference," *Foundations and Trends in Machine Learning*, vol. 9, no. 2-3, pp. 119–247, 2016.

[28] O. Simeone, *Machine Learning for Engineers*. Cambridge University Press, 2022.

[29] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight Uncertainty in Neural Network," in *International Conference on Machine Learning*. PMLR, 2015, pp. 1613–1622.

[30] S. Ravi and A. Beatson, "Amortized Bayesian Meta-Learning," in *Proc. International Conference on Learning Representations (ICLR), in Vancouver, Canada*, 2018, pp. 1–14.

[31] D. Barber, *Bayesian Reasoning and Machine Learning*. USA: Cambridge University Press, 2012.

[32] H. Wang and D.-Y. Yeung, "A Survey on Bayesian Deep Learning," *ACM Computing Surveys (CSUR)*, vol. 53, no. 5, pp. 1–37, 2020.

[33] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *International Conference on Machine Learning*. PMLR, 2016, pp. 1050–1059.

[34] A. Graves, "Practical Variational Inference for Neural Networks," *Proc. Advances in Neural Information Processing Systems (NIPS) in Granada, Spain*, vol. 24, 2011.

[35] M. Dusenberry, G. Jerfel, Y. Wen, Y. Ma, J. Snoek, K. Heller, B. Lakshminarayanan, and D. Tran, "Efficient and Scalable Bayesian Neural Nets with Rank-1 Factors," in *Proc. International Conference on Machine Learning in Baltimore , USA*. PMLR, 2020, pp. 2782–2792.

[36] E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig, "Laplace Redux-Effortless Bayesian Deep Learning," *Proc. Advances in Neural Information Processing Systems (NIPS) as Virtual-only Conference*, vol. 34, pp. 20 089–20 103, 2021.

[37] S. Farquhar, L. Smith, and Y. Gal, "Liberty or Depth: Deep Bayesian Neural Nets Do Not Need Complex Weight Posterior Approximations," *Proc. Advances in Neural Information Processing Systems (NIPS) as Virtual-only Conference*, vol. 33, pp. 4346–4357, 2020.

[38] Q. Liu and D. Wang, "Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm," *Proc. Advances in Neural Information Processing Systems (NIPS) in Barcelona, Spain*, vol. 29, 2016.

[39] M. Welling and Y. W. Teh, "Bayesian Learning via Stochastic Gradient Langevin Dynamics," in *Proceedings of the 28th International Conference on Machine learning (ICML-11) in Bellevue, Washington, USA*. Citeseer, 2011, pp. 681–688.

[40] R. Zhang, C. Li, J. Zhang, C. Chen, and A. G. Wilson, "Cyclical Stochastic Gradient MCMC for Bayesian Deep Learning," *Eighth International Conference on Learning Representations, Virtual Conference, Formerly Addis Ababa Ethiopia*, pp. 1–27, 2020.

[41] R. M. Neal, "MCMC Using Hamiltonian Dynamics," in *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC, 2011, pp. 139–188.

[42] S. Thrun, "Lifelong Learning Algorithms," in *Learning to Learn*. Springer, 1998, pp. 181–209.

[43] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70. PMLR, 06–11 Aug 2017, pp. 1126–1135.

[44] L. Zintgraf, K. Shiarli, V. Kurin, K. Hofmann, and S. Whiteson, "Fast Context Adaptation via Meta-Learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7693–7702.

[45] D. Maclaurin, D. Duvenaud, and R. Adams, "Gradient-Based Hyperparameter Optimization Through Reversible Learning," in *International conference on machine learning*. PMLR, 2015, pp. 2113–2122.

[46] Z. Li, F. Zhou, F. Chen, and H. Li, "Meta-SGD: Learning to Learn Quickly for Few-Shot Learning," *arXiv preprint arXiv:1707.09835*, 2017.

[47] H. S. Behl, A. G. Baydin, and P. H. Torr, "Alpha MAML: Adaptive Model-Agnostic Meta-Learning," *6th ICML Workshop on Automated Machine Learning (2019), Long Beach, California, USA*, pp. 1–11, 2019.

[48] J. Baxter, "Theoretical Models of Learning to Learn," in *Learning to learn*. Springer, 1998, pp. 71–94.

[49] S. Park, H. Jang, O. Simeone, and J. Kang, "Learning to Demodulate From Few Pilots via Offline and Online Meta-Learning," *IEEE Transactions on Signal Processing*, vol. 69, pp. 226–239, 2021.

[50] M. Goutay, F. Ait Aoudia, and J. Hoydis, "Deep HyperNetwork-Based MIMO Detection," in *Proc. IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC) in Atlanta, USA*, 2020, pp. 1–5.

[51] Y. Yuan, G. Zheng, K.-K. Wong, B. Ottersten, and Z.-Q. Luo, "Transfer Learning and Meta Learning-Based Fast Downlink Beamforming Adaptation," *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 1742–1755, 2021.

[52] Y. Hu, M. Chen, W. Saad, H. V. Poor, and S. Cui, "Meta-Reinforcement Learning for Trajectory Design in Wireless UAV Networks," in *Proc. GLOBECOM 2020- IEEE Global Communications Conference, in Taipei, Taiwan.* IEEE, 2020, pp. 1–6.

[53] T. Raviv, S. Park, N. Shlezinger, O. Simeone, Y. C. Eldar, and J. Kang, "Meta-ViterbiNet: Online Meta-Learned Viterbi Equalization for Non-Stationary Channels," in *Proc. 2021 IEEE International Conference on Communications Workshops (ICC Workshops) in Montreal, Canada.* IEEE, 2021, pp. 1–6.

[54] I. Nikoloska and O. Simeone, "Fast Power Control Adaptation via Meta-Learning for Random Edge Graph Neural Networks," in *Proc. 2021 IEEE 22nd International Workshop on Signal Processing Advances in Wireless Communications (SPAWC) in Lucca, Italy.* IEEE, 2021, pp. 146–150.

[55] J. Zhang, Y. Yuan, G. Zheng, I. Krikidis, and K.-K. Wong, "Embedding Model-Based Fast Meta Learning for Downlink Beamforming Adaptation," *IEEE Transactions on Wireless Communications*, vol. 21, no. 1, pp. 149–162, 2021.

[56] A. E. Kalør, O. Simeone, and P. Popovski, "Prediction of mmWave/THz Link Blockages through Meta-Learning and Recurrent Neural Networks," *IEEE Wireless Communications Letters*, vol. 10, no. 12, pp. 2815–2819, 2021.

[57] Y. Jiang, H. Kim, H. Asnani, and S. Kannan, "Mind: Model Independent Neural Decoder," in *Proc. 2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC) in Cannes, France.* IEEE, 2019, pp. 1–5.

[58] J. Zhang, Y. He, Y.-W. Li, C.-K. Wen, and S. Jin, "Meta Learning-Based MIMO Detectors: Design, Simulation, and Experimental Test," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1122–1137, 2020.

[59] C. Nguyen, T.-T. Do, and G. Carneiro, "Uncertainty in Model-Agnostic Meta-Learning using Variational Inference," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 3090–3100.

[60] K. Posch, J. Steinbrener, and J. Pilz, "Variational Inference to Measure Model Uncertainty in Deep Neural Networks," *arXiv preprint arXiv:1902.10189*, 2019.

[61] N. Houlsby, F. Huszár, Z. Ghahramani, and M. Lengyel, "Bayesian Active Learning for Classification and Preference Learning," *arXiv preprint arXiv:1112.5745*, 2011.

[62] Y. Gal, R. Islam, and Z. Ghahramani, "Deep Bayesian Active Learning with Image Data," in *International Conference on Machine Learning.* PMLR, 2017, pp. 1183–1192.

[63] F. Sohrabi, T. Jiang, W. Cui, and W. Yu, "Active Sensing for Communications by Learning," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 6, pp. 1780–1794, 2022.

[64] H. Sahbi, S. Deschamps, and A. Stoian, "Active Learning for Interactive Satellite Image Change Detection," *arXiv preprint arXiv:2110.04250*, 2021.

[65] J. Kaddour, S. Sæmundsson *et al.*, "Probabilistic Active Meta-Learning," *Proc. Advances in Neural Information Processing Systems (NIPS) as Virtual-only Conference*, vol. 33, pp. 20 813–20 822, 2020.

[66] A. Kirsch, J. Van Amersfoort, and Y. Gal, "BatchBALD: Efficient and Diverse Batch Acquisition for Deep Bayesian Active Learning," *Proc. Advances in Neural Information Processing Systems (NIPS) in Vancouver, Canada*, vol. 32, 2019.

[67] I. Nikoloska and O. Simeone, "Bayesian Active Meta-Learning for Black-Box Optimization," in *Proc. 2022 IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communications (SPAWC) in Oulu, Finland*. IEEE, 2022.

[68] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional Radio Modulation Recognition Networks," in *Engineering Applications of Neural Networks*. Cham: Springer International Publishing, 2016, pp. 213–226.

[69] T. J. O'Shea, T. Roy, and T. C. Clancy, "Over-The-Air Deep Learning Based Radio Signal Classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 168–179, 2018.

[70] X. Song, X. Fan, C. Xiang, Q. Ye, L. Liu, Z. Wang, X. He, N. Yang, and G. Fang, "A Novel Convolutional Neural Network Based Indoor Localization Framework with WiFi Fingerprinting," *IEEE Access*, vol. 7, pp. 110 698–110 709, 2019.

[71] N. Dvorecki, O. Bar-Shalom, L. Banin, and Y. Amizur, "A Machine Learning Approach for Wi-Fi RTT Ranging," in *Proceedings of the 2019 International Technical Meeting of the Institute of Navigation*, 2019, pp. 435–444.

[72] J. C. Platt, "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods," *Advances in Large Margin Classifiers*, pp. 61–74, 1999.

[73] B. Zadrozny and C. Elkan, "Transforming classifier Scores into Accurate Multiclass Probability Estimates," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 694–699.

[74] A. Kumar, P. S. Liang, and T. Ma, "Verified Uncertainty Calibration," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[75] X. Ma and M. B. Blaschko, "Meta-cal: Well-Controlled Post-hoc Calibration by Ranking," in *International Conference on Machine Learning*. PMLR, 2021, pp. 7235–7245.

[76] D. Stutz, K. D. Dvijotham, A. T. Cemgil, and A. Doucet, "Learning Optimal Conformal Classifiers," in *International Conference on Learning Representations*, 2021.

[77] B.-S. Einbinder, Y. Romano, M. Sesia, and Y. Zhou, "Training Uncertainty-Aware Classifiers with Conformalized Deep Learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 22 380–22 395, 2022.

[78] R. J. Tibshirani, R. Foygel Barber, E. Candes, and A. Ramdas, "Conformal Prediction under Covariate Shift," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[79] Y. Yang and A. K. Kuchibhotla, "Finite-Sample Efficient Conformal Prediction," *arXiv preprint arXiv:2104.13871*, 2021.

[80] A. Kumar, S. Sarawagi, and U. Jain, "Trainable Calibration Measures for Neural Networks from Kernel Mean Embeddings," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2805–2814.

[81] M. J. Holland, "Making Learning More Transparent using Conformalized Performance Prediction," *arXiv preprint arXiv:2007.04486*, 2020.

[82] A. Perez-Lebel, M. Le Morvan, and G. Varoquaux, "Beyond Calibration: Estimating the Grouping Loss of Modern Neural Networks," in *ICLR 2023–The Eleventh International Conference on Learning Representations*, 2023.

[83] I. Gibbs and E. Candes, "Adaptive Conformal Inference Under Distribution Shift," *Advances in Neural Information Processing Systems*, vol. 34, pp. 1660–1672, 2021.

[84] S. Feldman, L. Ringel, S. Bates, and Y. Romano, "Achieving Risk Control in Online Learning Settings," *Transactions on Machine Learning Research*, 2023.

[85] V. Vovk, G. Shafer, and I. Nouretdinov, "Self-Calibrating Probability Forecasting," *Advances in Neural Information Processing Systems*, vol. 16, 2003.

[86] V. Vovk and I. Petej, "Venn-Abers Predictors," in *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, 2014, pp. 829–838.

[87] A. Lambrou, I. Nouretdinov, and H. Papadopoulos, "Inductive Venn Prediction," *Annals of Mathematics and Artificial Intelligence*, vol. 74, pp. 181–201, 2015.

[88] H. Löfström, T. Löfström, U. Johansson, and C. Sönströd, "Calibrated Explanations: With Uncertainty Information and Counterfactuals," *Expert Systems with Applications*, vol. 246, p. 123154, 2024.

[89] I. C. Covert, S. Lundberg, and S.-I. Lee, "Explaining by Removing: A Unified Framework for Model Explanation," *The Journal of Machine Learning Research*, vol. 22, no. 1, pp. 9477–9566, 2021.

[90] 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, pp. 486–507, 2021.

[91] A. N. Angelopoulos, S. Bates, A. Fisch, L. Lei, and T. Schuster, "Conformal Risk Control," in *The Twelfth International Conference on Learning Representations*, 2024.

[92] S. Bates, A. Angelopoulos, L. Lei, J. Malik, and M. Jordan, "Distribution-Free, Risk-Controlling Prediction Sets," *Journal of the ACM (JACM)*, vol. 68, no. 6, pp. 1–34, 2021.

[93] J. Teneggi, M. Tivnan, W. Stayman, and J. Sulam, "How to Trust Your Diffusion Model: A Convex Optimization Approach to Conformal Risk Control," in *International Conference on Machine Learning*. PMLR, 2023, pp. 33 940–33 960.

[94] A. N. Angelopoulos, S. Bates, E. J. Candès, M. I. Jordan, and L. Lei, "Learn Then Test: Calibrating Predictive Algorithms to Achieve Risk Control," *arXiv preprint arXiv:2110.01052*, 2021.

[95] S. Flennerhag, Y. Schroecker, T. Zahavy, H. van Hasselt, D. Silver, and S. Singh, "Bootstrapped Meta-Learning," in *International Conference on Learning Representations*, 2022.

[96] K. Gopalakrishnan, S. K. Khaitan, A. Choudhary, and A. Agrawal, "Deep Convolutional Neural Networks with Transfer Learning for Computer Vision-Based Data-Driven Pavement Distress Detection," *Construction and building materials*, vol. 157, pp. 322–330, 2017.

[97] Z. Alyafeai, M. S. AlShaibani, and I. Ahmad, "A Survey on Transfer Learning in Natural Language Processing," *arXiv preprint arXiv:2007.04239*, 2020.

[98] P. G. Shivakumar and P. Georgiou, "Transfer Learning from Adult to Children for Speech Recognition: Evaluation, Analysis and Recommendations," *Computer speech & language*, vol. 63, p. 101077, 2020.

[99] Q. Yang, Y. Zhang, W. Dai, and S. J. Pan, *Transfer Learning*. Cambridge University Press, 2020.

[100] A. Nichol, J. Achiam, and J. Schulman, "On First-Order Meta-Learning Algorithms," *arXiv preprint arXiv:1803.02999*, 2018.

[101] S. Hochreiter, A. S. Younger, and P. R. Conwell, "Learning to Learn Using Gradient Descent," in *Artificial Neural Networks—ICANN 2001: International Conference Vienna, Austria, August 21–25, 2001 Proceedings 11*. Springer, 2001, pp. 87–94.

[102] S. Qiao, C. Liu, W. Shen, and A. L. Yuille, "Few-Shot Image Recognition by Predicting Parameters from Activations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7229–7238.

[103] G. Koch, R. Zemel, R. Salakhutdinov *et al.*, "Siamese Neural Networks for One-Shot Image Recognition," in *ICML deep learning workshop*, vol. 2, no. 1. Lille, 2015.

[104] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching Networks for One Shot Learning," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[105] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-Learning in Neural Networks: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5149–5169, 2022.

[106] J. MacInnes, S. Santosa, and W. Wright, "Visual Classification: Expert Knowledge Guides Machine Learning," *IEEE Computer Graphics and Applications*, vol. 30, no. 1, pp. 8–14, 2010.

[107] X. Zhang, Y.-C. Liang, and J. Fang, "Bayesian Learning Based Multiuser Detection for M2M Communications with Time-Varying User Activities," in *2017 IEEE International Conference on Communications (ICC) in Paris, France*. IEEE, 2017, pp. 1–6.

[108] ——, "Novel Bayesian Inference Algorithms for Multiuser Detection in M2M Communications," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 9, pp. 7833–7848, 2017.

[109] R. Prasad, C. R. Murthy, and B. D. Rao, "Joint Channel Estimation and Data Detection in MIMO-OFDM Systems: A Sparse Bayesian Learning Approach," *IEEE Transactions on Signal Processing*, vol. 63, no. 20, pp. 5369–5382, 2015.

[110] X. Lv, Y. Li, Y. Wu, X. Wang, and H. Liang, "Joint Channel Estimation and Impulsive Noise Mitigation Method for OFDM Systems Using Sparse Bayesian Learning," *IEEE Access*, vol. 7, pp. 74 500–74 510, 2019.

[111] K. M. Cohen, S. Park, O. Simeone, and S. Shamai Shitz, "Calibrating AI Models for Wireless Communications via Conformal Prediction," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 1, pp. 296–312, 2023.

[112] K. M. Cohen, S. Park, O. Simeone, and S. S. Shitz, "Calibrating AI Models for Few-Shot Demodulation via Conformal Prediction," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[113] K. M. Cohen, S. Park, O. Simeone, P. Popovski, and S. Shamai, "Guaranteed Dynamic Scheduling of Ultra-Reliable Low-Latency Traffic via Conformal Prediction," *IEEE Signal Processing Letters*, vol. 30, pp. 473–477, 2023.

[114] K. M. Cohen, S. Park, O. Simeone, and S. Shamai, "Learning to Learn to Demodulate with Uncertainty Quantification via Bayesian Meta-Learning," in *Proc. WSA 2021; 25th International ITG Workshop on Smart Antennas in EURECOM, France*. VDE, 2021, pp. 202–207.

[115] ——, "Cross-Validation Conformal Risk Control," in *accepted to IEEE International Symposium on Information Theory Proceedings (ISIT2024)*, July 2024.

[116] S. Park, K. M. Cohen, and O. Simeone, "Few-Shot Calibration of Set Predictors via Meta-Learned Cross-Validation-Based Conformal Prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 01, pp. 280–291, jan 2024.

[117] ——, "Few-Shot Calibration of Set Predictors via Meta-Learned Cross-Validation-Based Conformal Prediction," in *Sixth Workshop on Meta-Learning at the Conference on Neural Information Processing Systems*, 2022.

[118] S. Park, O. Bastani, J. Weimer, and I. Lee, "Calibrated Prediction with Covariate Shift via Unsupervised Domain Adaptation," *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

[119] J. Vaicenavicius, D. Widmann, C. Andersson, F. Lindsten, J. Roll, and T. Schön, "Evaluating Model Calibration in Classification," *Proc. 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 3459–3467, 2019.

[120] S. T. Jose and O. Simeone, "Information-Theoretic Generalization Bounds for Meta-Learning and Applications," *Entropy*, vol. 23, no. 1, p. 126, 2021.

[121] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT Press, 2016.

[122] O. Simeone, "A Brief Introduction to Machine Learning for Engineers," *Foundation and Trends in Signal Processing*, vol. 13, no. 12, Aug. 2018.

[123] R. Amit and R. Meir, "Meta-Learning by Adjusting Priors Based on Extended PAC-Bayes Theory," in *Proc. International Conference on Machine Learning (ICML) in Stockholm, Sweden*. PMLR, 2018, pp. 205–214.

[124] T. M. Cover and J. A. Thomas, "Information Theory and Statistics," *Elements of Information Theory*, vol. 1, no. 1, pp. 279–335, 1991.

[125] S. T. Jose and O. Simeone, "Free Energy Minimization: A Unified Framework for Modeling, Inference, Learning, and Optimization [lecture notes] ," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 120–125, 2021.

[126] A. Xu and M. Raginsky, "Information-Theoretic Analysis of Generalization Capability of Learning Algorithms," *Advances in neural information processing systems*, vol. 30, 2017.

[127] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[128] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths, "Recasting Gradient-Based Meta-Learning as Hierarchical Bayes," in *Sixth International Conference on Learning Representations (ICLR), in Vancouver, Canada*, 2018, pp. 1–14.

[129] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[130] A. Griewank, "Some Bounds on the Complexity of Gradients, Jacobians, and Hessians," in *Complexity in Numerical Optimization*. World Scientific, 1993, pp. 128–162.

[131] A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine, "Meta-Learning with Implicit Gradients," in *Proc. Neural Information Processing Systems (NIPS), in Vancouver, Canada*, 2019, pp. 113–124.

[132] M. Zecchin, S. Park, O. Simeone, M. Kountouris, and D. Gesbert, "Robust Bayesian Learning for Reliable Wireless AI: Framework and Applications," *IEEE Transactions on Cognitive Communications and Networking*, vol. 9, no. 4, pp. 897–912, 2023.

[133] S. Mohamed, M. Rosca, M. Figurnov, and A. Mnih, "Monte Carlo Gradient Estimation in Machine Learning," *Journal of Machine Learning Research (JMLR)*, vol. 21, no. 132, pp. 1–62, 2020.

[134] T. Melluish, C. Saunders, I. Nouretdinov, and V. Vovk, "Comparing the Bayes and Typicalness Frameworks," in *European Conference on Machine Learning (ECML) in Freiburg, Germany*. Springer, 2001, pp. 360–371.

[135] Y. Zhang, A. Doshi, R. Liston, W.-t. Tan, X. Zhu, J. G. Andrews, and R. W. Heath, "DeepWiPHY: Deep Learning-Based Receiver Design and Dataset for IEEE 802.11 ax systems," *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 1596–1611, 2020.

[136] A. G. Helmy, M. Di Renzo, and N. Al-Dhahir, "On the Robustness of Spatial Modulation to I/Q Imbalance," *IEEE Communications Letters*, vol. 21, no. 7, pp. 1485–1488, 2017.

[137] D. Tandur and M. Moonen, "Joint Adaptive Compensation of Transmitter and Receiver IQ Imbalance under Carrier Frequency Offset in OFDM-Based Systems," *IEEE Transactions on Signal Processing*, vol. 55, no. 11, pp. 5246–5252, 2007.

[138] M. Yin, G. Tucker, M. Zhou, S. Levine, and C. Finn, "Meta-Learning without Memorization," in *International Conference on Learning Representations*, 2020, pp. 1–21.

[139] O. Narmanlioglu, E. Zeydan, M. Kandemir, and T. Kranda, "Prediction of Active UE Number with Bayesian Neural Networks for Self-Organizing LTE Networks," in *2017 8th International Conference on the Network of the Future (NOF)*. IEEE, 2017, pp. 73–78.

[140] Z. Wu and H. Li, "Stochastic Gradient Langevin Dynamics for Massive MIMO Detection," *IEEE Communications Letters*, vol. 26, no. 5, pp. 1062–1065, 2022.

[141] N. Zilberstein, C. Dick, R. Doost-Mohammady, A. Sabharwal, and S. Segarra, "Annealed Langevin Dynamics for Massive MIMO Detection," *IEEE Transactions on Wireless Communications*, vol. 22, no. 6, pp. 3762–3776, 2023.

[142] Z. Tao and S. Wang, "Improved Downlink Rates for FDD Massive MIMO Systems through Bayesian Neural Networks-Based Channel Prediction," *IEEE Transactions on Wireless Communications*, vol. 21, no. 3, pp. 2122–2134, 2021.

[143] N. K. Jha and V. K. Lau, "Transformer-Based Online Bayesian Neural Networks for Grant-Free Uplink Access in CRAN With Streaming Variational Inference," *IEEE Internet of Things Journal*, vol. 9, no. 9, pp. 7051–7064, 2021.

[144] ——, "Online Downlink Multi-User Channel Estimation for mmWave Systems using Bayesian Neural Network," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2374–2387, 2021.

[145] J. Xu, Y. Shen, E. Chen, and V. Chen, "Bayesian Neural Networks for Identification and Classification of Radio Frequency Transmitters using Power Amplifiers' Nonlinearity Signatures," *IEEE Open Journal of Circuits and Systems*, vol. 2, pp. 457–471, 2021.

[146] D. Liu, G. Zhu, Q. Zeng, J. Zhang, and K. Huang, "Wireless Data Acquisition for Edge Learning: Data-Importance Aware Retransmission," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 406–420, 2020.

[147] S.-E. Chiu, N. Ronquillo, and T. Javidi, "Active Learning and CSI Acquisition for mmWave Initial Alignment," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 11, pp. 2474–2489, 2019.

[148] K. Yang, J. Ren, Y. Zhu, and W. Zhang, "Active Learning for Wireless IoT Intrusion Detection," *IEEE Wireless Communications*, vol. 25, no. 6, pp. 19–25, 2018.

[149] M. K. Abdel-Aziz, S. Samarakoon, M. Bennis, and W. Saad, "Ultra-Reliable and Low-Latency Vehicular Communication: An Active Learning Approach," *IEEE Communications Letters*, vol. 24, no. 2, pp. 367–370, 2019.

[150] C. Finn, K. Xu, and S. Levine, "Probabilistic Model-Agnostic Meta-Learning," *Proc. Advances in Neural Information Processing Systems (NIPS) in Montreal, Canada*, vol. 31, 2018.

[151] J. Yoon, T. Kim, O. Dia, S. Kim, Y. Bengio, and S. Ahn, "Bayesian Model-Agnostic Meta-Learning," *Proc. Advances in Neural Information Processing Systems (NIPS), in Montreal, Canada*, vol. 31, 2018.

[152] M. Patacchiola, J. Turner, E. J. Crowley, M. O'Boyle, and A. J. Storkey, "Bayesian Meta-Learning for the Few-Shot Setting via Deep Kernels," *Proc. Advances in Neural Information Processing Systems (NIPS) as Virtual-only Conference*, vol. 33, pp. 16 108–16 118, 2020.

[153] Y. Zou and X. Lu, "Gradient-EM Bayesian Meta-Learning," *Proc. Advances in Neural Information Processing Systems (NIPS) as Virtual-only Conference*, vol. 33, pp. 20 865–20 875, 2020.

[154] J. Rothfuss, V. Fortuin, M. Josifoski, and A. Krause, "PACOH: Bayes-Optimal Meta-Learning with PAC-Guarantees," in *(ICML), as Virtual-only Conference*. PMLR, 2021, pp. 9116–9126.

[155] J. Rothfuss, D. Heyn, A. Krause *et al.*, "Meta-Learning Reliable Priors in the Function Space," *Proc. Advances in Neural Information Processing Systems (NIPS) as Virtual-only Conference*, vol. 34, pp. 280–293, 2021.

[156] S. T. Jose, S. Park, and O. Simeone, "Information-Theoretic Analysis of Epistemic Uncertainty in Bayesian Meta-learning," in *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS) as a Virtual-only Conference*. PMLR, 2022, pp. 9758–9775.

[157] I. Nikoloska and O. Simeone, "Quantum-Aided Meta-Learning for Bayesian Binary Neural Networks via Born Machines," in *2022 IEEE 32nd International Workshop on Machine Learning for Signal Processing (MLSP)*, 2022, pp. 1–6.

[158] B. de Finetti, "Theory of Probability, vol. 1," 1974.

[159] R. Koenker and G. Bassett Jr, "Regression Quantiles," *Econometrica: Journal of the Econometric Society*, pp. 33–50, 1978.

[160] I. Steinwart and A. Christmann, "Estimating Conditional Quantiles with the Help of the Pinball Loss," *Bernoulli*, vol. 17, no. 1, pp. 211–225, 2011.

[161] 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, pp. 1094–1111, 2018.

[162] R. F. Barber, E. J. Candes, A. Ramdas, and R. J. Tibshirani, "Conformal Prediction Beyond Exchangeability," *The Annals of Statistics*, vol. 51, no. 2, pp. 816–845, 2023.

[163] Y. Romano, M. Sesia, and E. Candes, "Classification with Valid and Adaptive Coverage," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3581–3591, 2020.

[164] Z. Wang, R. Gao, M. Yin, M. Zhou, and D. Blei, "Probabilistic Conformal Prediction Using Conditional Random Samples," in *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, F. Ruiz, J. Dy, and J.-W. van de Meent, Eds., vol. 206. PMLR, 25–27 Apr 2023, pp. 8814–8836.

[165] M. Zaffran, O. Féron, Y. Goude, J. Josse, and A. Dieuleveut, "Adaptive Conformal Predictions for Time Series," in *International Conference on Machine Learning*. PMLR, 2022, pp. 25 834–25 866.

[166] Z. Lin, S. Trivedi, and J. Sun, "Conformal Prediction with Temporal Quantile Adjustments," *Advances in Neural Information Processing Systems*, vol. 35, pp. 31 017–31 030, 2022.

[167] N. Seshadri and C.-E. Sundberg, "List Viterbi Decoding Algorithms with Applications," *IEEE Transactions on Communications*, vol. 42, no. 234, pp. 313–323, 1994.

[168] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[169] F. Wenzel, K. Roth, B. S. Veeling, J. Swiatkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin, "How Good is the Bayes Posterior in Deep Neural Networks Really?" in *Proceedings of the 37th International Conference on Machine Learning*, H. Daumé III and A. Singh, Eds., vol. 119. Cambridge, MA: PMLR, 2020, Conference Paper, pp. 10 248 – 10 259.

[170] N. Ye, Z. Zhu, and R. K. Mantiuk, "Langevin Dynamics with Continuous Tempering for Training Deep Neural Networks," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 618–626.

[171] Z. Zhu and A. K. Nandi, *Automatic Modulation Classification: Principles, Algorithms and Applications*. John Wiley & Sons, 2015.

[172] R. Zhou, F. Liu, and C. W. Gravelle, "Deep Learning for Modulation Recognition: A Survey with a Demonstration," *IEEE Access*, vol. 8, pp. 67 366–67 376, 2020.

[173] ETSI, "Reconfigurable Radio Systems (RRS); Feasibility Study on Radio Frequency (RF) Performance for Cognitive Radio Systems Operating in UHF TV Band White Spaces," ETSI, Tech. Rep. TR 103 067 V1.1.1 (2013-05), 2013.

[174] P. Aparna and M. Jayasheela, "Cyclostationary Feature Detection in Cognitive Radio using Different Modulation Schemes," *International Journal of Computer Applications*, vol. 47, no. 21, 2012.

[175] J. Nieminen, R. Jantti, and L. Qian, "Time Synchronization of Cognitive Radio Networks," in *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*. IEEE, 2009, pp. 1–6.

[176] Z. Kollar and P. Horvath, "Physical Layer Considerations for Cognitive Radio: Synchronization Point of View," in *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)*. IEEE, 2011, pp. 1–5.

[177] I. C. Wong and B. L. Evans, "Optimal Resource Allocation in the OFDMA Downlink with Imperfect Channel Knowledge," *IEEE Transactions on Communications*, vol. 57, no. 1, pp. 232–241, 2009.

[178] I. Nikoloska and O. Simeone, "Modular Meta-Learning for Power Control via Random Edge Graph Neural Networks," *IEEE Transactions on Wireless Communications*, 2022.

[179] H. Shokri-Ghadikolaei, C. Fischione, G. Fodor, P. Popovski, and M. Zorzi, "Millimeter Wave Cellular Networks: A MAC Layer Perspective," *IEEE Transactions on Communications*, vol. 63, no. 10, pp. 3437–3458, 2015.

[180] L. Maggi, A. Valcarce, and J. Hoydis, "Bayesian Optimization for Radio Resource Management: Open Loop Power Control," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, p. 1858–1871, Jul 2021.

[181] Y. Zhang, O. Simeone, S. T. Jose, L. Maggi, and A. Valcarce, "Bayesian and Multi-Armed Contextual Meta-Optimization for Efficient Wireless Radio Resource Management," *IEEE Transactions on Cognitive Communications and Networking*, vol. 9, no. 5, pp. 1282–1295, 2023.

[182] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[183] C. Cox, *An Introduction to 5G: The New Radio, 5G Network and Beyond*. John Wiley & Sons, 2020.

[184] P. Popovski, K. F. Trillingsgaard, O. Simeone, and G. Durisi, "5G Wireless Network Slicing for eMBB, URLLC, and mMTC: A Communication-Theoretic View," *IEEE Access*, vol. 6, pp. 55 765–55 779, 2018.

[185] R. Vannithamby and A. Soong, *5G Verticals: Customizing Applications, Technologies and Deployment Techniques*. John Wiley & Sons, 2020.

[186] A. Anand, G. De Veciana, and S. Shakkottai, "Joint Scheduling of URLLC and eMBB Traffic in 5G Wireless Networks," *IEEE/ACM Transactions on Networking*, vol. 28, no. 2, pp. 477–490, 2020.

[187] R. Kassab, O. Simeone, and P. Popovski, "Coexistence of URLLC and eMBB Services in the C-RAN Uplink: An Information-Theoretic Study," in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–6.

[188] A. A. Esswie and K. I. Pedersen, "Opportunistic Spatial Preemptive Scheduling for URLLC and eMBB Coexistence in Multi-User 5G Networks," *IEEE Access*, vol. 6, pp. 38 451–38 463, 2018.

[189] P. C. Eggers, M. Angjelichinoski, and P. Popovski, "Wireless Channel Modeling Perspectives for Ultra-Reliable Communications," *IEEE Transactions on Wireless Communications*, vol. 18, no. 4, pp. 2229–2243, 2019.

[190] M. Angjelichinoski, K. F. Trillingsgaard, and P. Popovski, "A Statistical Learning Approach to Ultra-Reliable Low Latency Communication," *IEEE Transactions on Communications*, vol. 67, no. 7, pp. 5153–5166, 2019.

[191] M. Alsenwi, N. H. Tran, M. Bennis, A. K. Bairagi, and C. S. Hong, "eMBB-URLLC Resource Slicing: A Risk-Sensitive Approach," *IEEE Communications Letters*, vol. 23, no. 4, pp. 740–743, 2019.

[192] A. Anand and G. de Veciana, "Resource Allocation and HARQ Optimization for URLLC Traffic in 5G Wireless Networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 11, pp. 2411–2421, 2018.

[193] T. Ma, Y. Zhang, F. Wang, D. Wang, and D. Guo, "Slicing Resource Allocation for eMBB and URLLC in 5G RAN," *Wireless Communications and Mobile Computing*, vol. 2020, pp. 1–11, 2020.

[194] N. H. Mahmood, O. A. Lopez, H. Alves, and M. Latva-Aho, "A Predictive Interference Management Algorithm for URLLC in Beyond 5G Networks," *IEEE Communications Letters*, vol. 25, no. 3, pp. 995–999, 2020.

[195] C. Padilla, R. Hashemi, N. H. Mahmood, and M. Latva-Aho, "A Nonlinear Autoregressive Neural Network for Interference Prediction and Resource Allocation in URLLC Scenarios," in *2021 International Conference on Information and Communication Technology Convergence (ICTC)*, 2021, pp. 184–189.

[196] H. Khan, M. M. Butt, S. Samarakoon, P. Sehier, and M. Bennis, "Deep Learning Assisted CSI Estimation for Joint URLLC and eMBB Resource Allocation," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2020, pp. 1–6.

[197] C. Sun and C. Yang, "Learning to Optimize with Unsupervised Learning: Training Deep Neural Networks for URLLC," in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2019, pp. 1–7.

[198] J. Zhang, C. Sun, and C. Yang, "Resource Allocation in URLLC with Online Learning for Mobile Users," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1–5.

[199] M. Alsenwi, N. H. Tran, M. Bennis, S. R. Pandey, A. K. Bairagi, and C. S. Hong, "Intelligent Resource Slicing for eMBB and URLLC Coexistence in 5G and Beyond: A Deep Reinforcement Learning Based Approach," *IEEE Transactions on Wireless Communications*, vol. 20, no. 7, pp. 4585–4600, 2021.

[200] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic Learning in a Random World, Second Edition*. Springer International Publishing, Jan 2022, publisher Copyright: © Springer Verlag New York, Inc. 2005.

[201] G. Shafer and V. Vovk, "A Tutorial on Conformal Prediction," *Journal of Machine Learning Research*, vol. 9, no. 3, 2008.

[202] L. Gyôrfi and H. Walk, "Nearest Neighbor Based Conformal Prediction," in *Annales de l'ISUP*, vol. 63, no. 2-3, 2019, pp. 173–190.

[203] C. Lu, A. Lemay, K. Chang, K. Höbel, and J. Kalpathy-Cramer, "Fair Conformal Predictors for Applications in Medical Imaging," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 11. PMLR, 2022, pp. 12 008–12 016.

[204] C. Lu, K. Chang, P. Singh, and J. Kalpathy-Cramer, "Three Applications of Conformal Prediction for Rating Breast Density in Mammography," *arXiv preprint arXiv:2206.12008*, 2022.

[205] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe Planning in Dynamic Environments Using Conformal Prediction," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 5116–5123, 2023.

[206] L. Andéol, T. Fel, F. De Grancey, and L. Mossina, "Conformal Prediction for Trustworthy Detection of Railway Signals," 2023.

[207] C. Xu and Y. Xie, "Conformal Prediction Interval for Dynamic Time-Series," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 18–24 Jul 2021, pp. 11 559–11 569.

[208] R. Kassab, O. Simeone, P. Popovski, and T. Islam, "Non-Orthogonal Multiplexing of Ultra-Reliable and Broadband Services in Fog-Radio Architectures," *IEEE Access*, vol. 7, pp. 13 035–13 049, 2019.

[209] S. Cavallero, N. S. Grande, F. Pase, M. Giordani, J. Eichinger, R. Verdone, and M. Zorzi, "A New Scheduler for URLLC in 5G NR IIoT Networks with Spatio-Temporal Traffic Correlations," in *2017 IEEE International Conference on Communications (ICC) in Rome, Italy*. IEEE, 2023.

[210] S. Cammerer, T. Gruber, J. Hoydis, and S. Ten Brink, "Scaling Deep Learning-Based Decoding of Polar Codes via Partitioning," in *GLOBECOM 2017-2017 IEEE global communications conference*. IEEE, 2017, pp. 1–6.

[211] N. Shlezinger, R. Fu, and Y. C. Eldar, "DeepSIC: Deep Soft Interference Cancellation for Multiuser MIMO Detection," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1349–1362, 2020.

[212] R. M. Gray and R. Gray, *Probability, Random Processes, and Ergodic Properties*. Springer, 2009, vol. 1.

[213] D. Tran, J. Z. Liu, M. W. Dusenberry, D. Phan, M. Collier, J. Ren, K. Han, Z. Wang, Z. E. Mariet, H. Hu, N. Band, T. G. J. Rudner, Z. Nado, J. van Amersfoort, A. Kirsch, R. Jenatton, N. Thain, E. K. Buchanan, K. P. Murphy, D. Sculley, Y. Gal, Z. Ghahramani, J. Snoek, and B. Lakshminarayanan, "Plex: Towards Reliability using Pretrained Large Model Extensions," in *First Workshop on Pre-training: Perspectives, Pitfalls, and Paths Forward at ICML 2022*, 2022.

[214] B. Rajendran, O. Simeone, and B. M. Al-Hashimi, "Towards Efficient and Trustworthy AI Through Hardware-Algorithm-Communication Co-Design," *arXiv preprint arXiv:2309.15942*, 2023.

[215] J. Lei and L. Wasserman, "Distribution-Free Prediction bands for non-parametric Regression," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 76, no. 1, pp. 71–96, 2014.

[216] N. Deutschmann, M. Rigotti, and M. R. Martinez, "Adaptive Conformal Regression with Jackknife+ Rescaled Scores," *arXiv preprint arXiv:2305.19901*, 2023.

[217] C. Gupta, A. K. Kuchibhotla, and A. Ramdas, "Nested Conformal Prediction and Quantile Out-Of-Bag Ensemble Methods," *Pattern Recognition*, vol. 127, p. 108496, 2022.

[218] S. Park and O. Simeone, "Quantum Conformal Prediction for Reliable Uncertainty Quantification in Quantum Machine Learning," *IEEE Transactions on Quantum Engineering*, no. 01, pp. 1–24, nov 2023.

[219] Y. Zhang, S. Park, and O. Simeone, "Bayesian Optimization with Formal Safety Guarantees via Online Conformal Prediction," *arXiv preprint arXiv:2306.17815*, 2023.

[220] A. N. Angelopoulos, S. Bates, M. I. Jordan, and J. Malik, "Uncertainty Sets for Image Classifiers using Conformal Prediction," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[221] Y. Romano, E. Patterson, and E. Candes, "Conformalized Quantile Regression," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[222] T. Omi, N. Ueda, and K. Aihara, "Fully Neural Network Based Model for General Temporal Point Processes," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, pp. 2122–2132.

[223] M. Dubey, R. Palakkadavath, and P. Srijith, "Bayesian Neural Hawkes Process for Event Uncertainty Prediction," *International Journal of Data Science and Analytics*, pp. 1–15, 2023.

[224] A. G. Hawkes, "Spectra of Some Self-Exciting and Mutually Exciting Point Processes," *Biometrika*, vol. 58, no. 1, pp. 83–90, 1971.

[225] R. Li, O. Bohdal, R. Mishra, H. Kim, D. Li, N. Lane, and T. Hospedales, "A Channel Coding Benchmark for Meta-Learning," in *Proc. Advances in Neural Information Processing Systems (NIPS) as a Virtual-only Conference, Track on Datasets and Benchmarks*, 2021.

[226] S. Ravi and H. Larochelle, "Meta-Learning for Batch Mode Active Learning," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018.

[227] A. Farinhas, C. Zerva, D. T. Ulmer, and A. Martins, "Non-Exchangeable Conformal Risk Control," in *The Twelfth International Conference on Learning Representations*, 2024.

[228] J. Chen, L. Song, M. Wainwright, and M. Jordan, "Learning to Explain: An Information-Theoretic Perspective on Model Interpretation," in *International conference on machine learning*. PMLR, 2018, pp. 883–892.

[229] A. Alkhatib, H. Bostrom, S. Ennadir, and U. Johansson, "Approximating Score-based Explanation Techniques Using Conformal Regression," in *Conformal and Probabilistic Prediction with Applications*. PMLR, 2023, pp. 450–469.

[230] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS 2017 Workshop on Autodiff*, 2017.

[231] J. Knoblauch, J. Jewson, and T. Damoulas, "Generalized Variational Inference: Three Arguments for Deriving New Posteriors," *arXiv preprint arXiv:1904.02063*, 2019.

[232] C. Finn, A. Rajeswaran, S. Kakade, and S. Levine, "Online Meta-Learning," in *International Conference on Machine Learning (ICML) in Long Beach, CA, USA*. PMLR, 2019, pp. 1920–1930.

[233] S. Feldman, S. Bates, and Y. Romano, "Conformalized Online Learning: Online Calibration Without a Holdout Set," *arXiv preprint arXiv:2205.09095*, 2022.

[234] A. K. Kuchibhotla, "Exchangeability, Conformal Prediction, and Rank Tests," *arXiv preprint arXiv:2005.06095*, 2020.

[235] A. Dean and J. Verducci, "Linear Transformations that Preserve Majorization, Schur Concavity, and Exchangeability," *Linear Algebra and Its Applications*, vol. 127, pp. 121–138, 1990.