eman ta zabal zazu

Universidad del País Vasco

Euskal Herriko Unibertsitatea

PhD Thesis

Supervised Learning in Time-dependent Environments with Performance Guarantees

Verónica Álvarez

Supervised by

Santiago Mazuelas

Jose A. Lozano

Donostia - San Sebastián, 2023

PhD Thesis

Supervised Learning in Time-dependent Environments with Performance Guarantees

Verónica Álvarez

Supervised by

Santiago Mazuelas

Jose A. Lozano

Donostia - San Sebastián, 2023

Funding in direct support of this work has been provided by the Basque Government through the ELKARTEK, the Spanish Ministry of Science, Innovation and Universities through BCAM's Severo Ochoa Excellence Accreditation SEV-2017-0718 and Project PID2019-105058GA-I00, the Spanish Ministry of Economic Affairs and Digital Transformation through Project IA4TES MIA.2021.M04.0008, and , IT-1504-22, and BERC 2022-2025 programmes. Verónica Álvarez has been supported by a grant of the University of the Basque Country UPV/EHU (PIFI20/02).

Acknowledgement

I would like to express my sincere gratitude to my advisors, Santiago Mazuelas and Jose Antonio Lozano, for their unwavering support and guidance throughout my Ph.D. journey. Their dedication and assistance have been invaluable in helping me overcome the challenges I encountered while writing each chapter of this dissertation. I am truly grateful for the time, patience, and motivation they have generously provided. I am fortunate to have had such exceptional advisors and mentors during my Ph.D. thesis and the early stages of my research career.

Besides my advisors, I would like to extend my thanks to Sanjoy Dasgupta for providing me with the invaluable opportunity to be a visiting researcher in the Department of Computer Science and Engineering at the University of California, San Diego.

I would also like to thank all my colleagues at BCAM for the stimulating discussions, idea exchanges, and experiences that we have shared throughout this time. I am truly fortunate to have had such an exceptional group of Ph.D. mates. Their friendship, encouragement, and unwavering support have made this journey unforgettable. I am indebted to each and every one of them for their contributions to my research and for making this Ph.D. experience truly remarkable.

Lastly, but certainly not least, I want to express my deepest gratitude to my family and friends for their unwavering support in every decision I have made. I am profoundly grateful for their constant encouragement and support throughout my college career. Their love and sacrifices have been the driving force behind my academic achievements. I am thankful for always being there for me, regardless of the circumstances, and for being a constant source of inspiration in my life.

Abstract

In practical scenarios, it is common to learn from a sequence of related problems (tasks). Such tasks are usually time-dependent in the sense that consecutive tasks are often significantly more similar. Time-dependency is common in multiple applications such as load forecasting, spam main filtering, and face emotion recognition. For instance, in the problem of load forecasting, the consumption patterns in consecutive time periods are significantly more similar since human habits and weather factors change gradually over time. Learning from a sequence tasks holds promise to enable accurate performance even with few samples per task by leveraging information from different tasks. However, harnessing the benefits of learning from a sequence of tasks is challenging since tasks are characterized by different underlying distributions.

Most existing techniques are designed for situations where the tasks' similarities do not depend on their order in the sequence. Existing techniques designed for time-dependent tasks adapt to changes between consecutive tasks accounting for a scalar rate of change by using a carefully chosen parameter such as a learning rate or a weight factor. However, the tasks' changes are commonly multidimensional, i.e., the time-dependency often varies across different statistical characteristics describing the tasks. For instance, in the problem of load forecasting, the statistical characteristics related to weather factors often change differently from those related to generation.

In this dissertation, we establish methodologies for supervised learning from a sequence of time-dependent tasks that effectively exploit information from all tasks, provide multidimensional adaptation to tasks' changes, and provide computable tight performance guarantees. We develop methods for supervised learning settings where tasks arrive over time including techniques for supervised classification under concept drift (SCD) and techniques for continual learning (CL). In addition, we present techniques for load forecasting that can adapt to time changes in consumption patterns and assess intrinsic uncertainties in load demand. The numerical results show that the proposed methodologies can significantly improve the performance of existing methods using multiple benchmark datasets. This dissertation makes theoretical contributions leading to efficient algorithms for multiple machine learning scenarios that provide computable performance guarantees and superior performance than state-of-the-art techniques.

Contents

Chapter 0

Introduction

In practical scenarios, it is often of interest to learn from a sequence of problems (tasks). Such scenarios include multi-source domain adaptation (MDA) (e.g., [1, 2]), multi-task learning (MTL) (e.g., [3, 4]), supervised classification under concept drift (SCD) (e.g., [5, 6]), and continual learning (CL) (e.g., [7, 8]). For instance, MDA has been used to learn to recognize diseases in medical data using medical images from different patients [9], and MTL has been used to learn to recognize facial emotions using images taken from multiple facial expressions [10].

Learning from a sequence of tasks can increase the performance of each task using information from all the tasks in the sequence [7,8,11]. Such transfer of information can enable accurate classification even in cases with reduced sample sizes, thus significantly increasing the effective sample size (ESS) of each task. However, exploiting the benefits of learning from a sequence of tasks is challenging since tasks are characterized by different underlying distributions [12–16].

Tasks in a sequence are usually time-dependent in the sense that consecutive tasks often have a higher similarity. Time-dependency is common in multiple applications including load forecasting [17], spam mail filtering [18], portraits classification [19], and credit card fraud detection [20]. For instance, in the problem of classification of portraits from different time periods [19]; the similarity between consecutive tasks (portraits of consecutive time periods) is significantly higher (see Figure 1).

0.1 Learning from a sequence of tasks

This section briefly describes machine learning scenarios for supervised learning from a sequence of tasks. Such scenarios can be classified as batch learning scenarios and online learning scenarios. Batch learning uses a pool of training samples to obtain classification rules and includes MDA and MTL; while online learning uses training samples that arrive over time to obtain classification rules at each time step and includes online adaptive learning and CL.

MDA learns the last task in the sequence (target domain) using information from

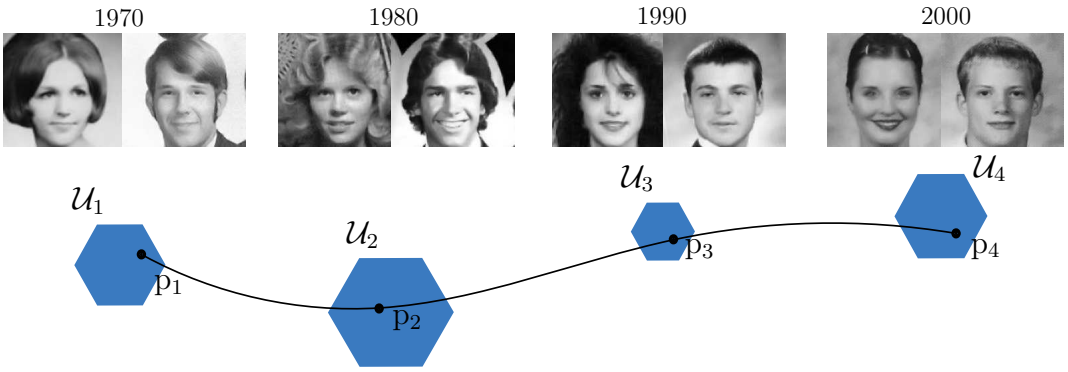


Figure 1: Tasks in a sequence are characterized by different underlying distributions and consecutive tasks are often more similar. The proposed learning methodology obtains an uncertainty set $\mathcal{U}_i$ for each task that can include the underlying distribution p_i.

different tasks (source domains) [1, 2, 21–29], while MTL simultaneously learns the sequence of tasks using information from all the tasks in the sequence [3, 4, 30–37]. For instance, MDA can be used to classify portraits of men and women from the 80's using portrait photographs from the 60's and 70's (source tasks), while MTL can be used to classify portraits from the 60's, 70's, and 80's using portrait photographs from such time periods. MDA is common in multiple applications such as sentiment classification [38], webpage tagging [39], and face recognition [40]; and MTL is common in multiple application such as face emotion recognition [10], object recognition [41], and animals classification [42].

In online adaptive learning and CL, sample sets corresponding with different tasks arrive over time. Online adaptive learning learns at each time step the last task using information from preceding tasks [5, 6, 43–48], while CL learns at each time step the sequence of tasks using information from all observed tasks [7, 8, 11, 14–16, 49–53]. For instance, online adaptive learning and CL can be used to classify portraits of men and women over time. Online adaptive learning can be used to classify at the 80's portraits from the 80's using portrait photographs up to the 70's, and CL can be used to classify at the 80's portraits from the 60's, 70's, and 80's using portraits photographs from such time periods. Online adaptive learning is common in multiple applications including regression problems such as load forecasting [17] and weather prediction [44] and classification problems (classification in these settings in commonly referred to as SCD) such as spam mail filtering [18] and credit card fraud detection [20]; and CL is common in multiples applications such as robotics [54] and human activity recognition [55].

0.2 Existing methods

In this section, we briefly review methods that have been used for MDA, SCD, MTL, and CL.

Techniques developed for MDA learn the target task by using information from source tasks [1,2,21–29]. Ensemble approaches use a weighted combination of information (e.g., samples and classification rules) from source tasks [1,24,25], and distribution alignment approaches learn the target task by using feature representations that align source and target distributions [26–29]. For instance, Muandet et al. [27] learn the target task by obtaining task invariant features that are shared across all tasks.

Techniques developed for MTL learn the whole sequence of tasks by using information from all the tasks [3, 4, 30–37]. Soft parameter sharing approaches learn specific layers for each task [32, 33], while hard parameter sharing approaches learn common hidden layers across all tasks [34–36]. For instance, Long et al. [35] learn five common convolutional layers of AlexNet [56] across all tasks.

Techniques developed for SCD learn at each time step the last task in the sequence by using information from the most recent tasks [5,6,43–48]. Sliding window approaches use a set of stored samples from the most recent preceding tasks [57,58], while learning rate approaches slightly change the classification rule for the preceding task using the samples from the most recent task [46, 48]. For instance, Shen et al. [48] utilize a time-varying combination of rules obtained with different learning rates.

Techniques developed for CL learn at each time step the whole sequence of observed tasks by using information from all the tasks [7, 8, 11, 14–16, 49–53]. Dynamic architecture approaches learn shared parameters using samples from all the tasks together with task-specific parameters using samples from the corresponding task [14, 51]; and replay approaches learn parameters using a pool of stored samples from all the preceding tasks together with the samples from the last task [11, 16, 59]. For instance, Lopez-Paz and Ranzato [11] solve at each time step an optimization problem with constraints determined by samples from preceding tasks.

Existing techniques for MDA, MTL, and CL are designed for situations where the tasks' similarities do not depend on their order in the sequence and cannot capture the usual time-dependency. In the current literature, only techniques for SCD [46–48, 57, 60–63] and the work for CL presented in [64] are designed for time-dependent tasks. Such methods adapt to changes between consecutive tasks accounting for a scalar rate of change by using a carefully chosen parameter such as a learning rate or a weight factor. However, the tasks' changes cannot be adequately addressed accounting only for a scalar rate of change. Such inadequacy is due to the fact that changes are commonly multidimensional, i.e., different statistical characteristics describing the tasks often change in a different manner.

Existing techniques do not provide computable performance guarantees in time-dependent environments. Techniques for MDA can provide non-computable performance guarantees in terms of distribution discrepancies between the source tasks and the target task [1, 65–67], while current techniques for MTL can provide computable performance guarantees based on the PAC-Bayes theory but are not designed for time-dependent tasks [68–70]. In addition, techniques for SCD can provide non-computable performance guarantees for time-dependent tasks in terms of discrepancies between consecutive distributions [71, 72]; while in the current literature of CL, only [64] provides

performance guarantees for time-dependent tasks based on PAC-Bayes theory but are non-computable in practice.

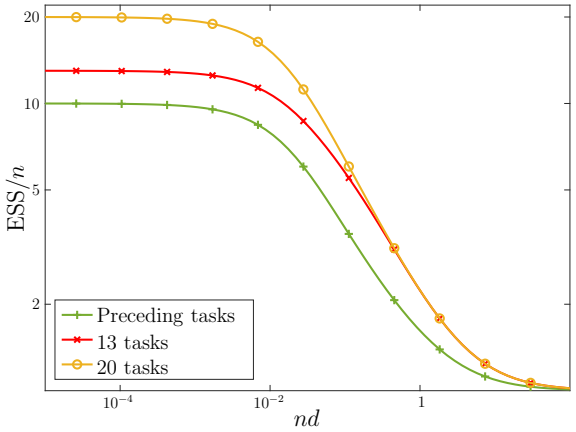
0.3 Contributions

This section describes the technical contributions, summarizes the results, and provides an outline of the dissertation.

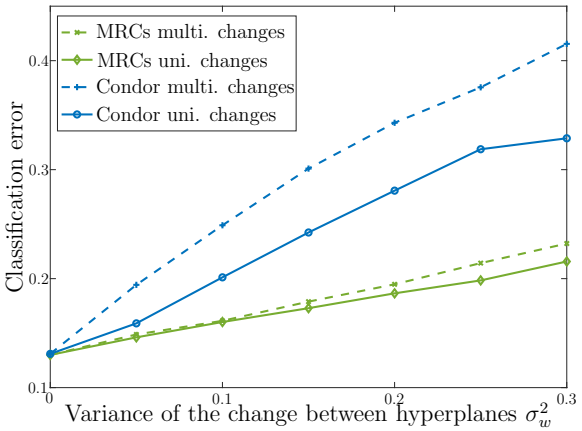
In this dissertation, we establish methodologies for supervised learning from a sequence of time-dependent tasks that effectively exploit information from all tasks, provide multidimensional adaptation to tasks' changes, and provide computable tight performance guarantees. We develop methods for supervised learning settings where tasks arrive over time including techniques for SCD and techniques for CL. In addition, we present techniques for load forecasting that can adapt to time changes in consumption patterns and assess intrinsic uncertainties in load demand. The numerical results show that the proposed techniques can significantly improve the performance of existing methods using multiple benchmark datasets. This dissertation makes theoretical contributions leading to efficient algorithms for multiple machine learning scenarios that provide computable performance guarantees and superior performance than state-of-the art techniques.

The proposed learning methodologies for classification sequentially obtain rules with the smallest worst-case expected loss over distributions in an uncertainty set. Such techniques efficiently leverage information from all tasks in the sequence and can harness changes in tasks' distributions by using a sequence of uncertainty sets that can contain the true underlying distributions (see Figure 1). The experimental results assess the performance of the proposed methodologies in comparison with existing techniques and the reliability of the presented bounds.

The contributions of this thesis are organized in 4 chapters corresponding to 4 papers: "Minimax classification under concept drift with multidimensional adaptation and performance guarantees," published in the *International Conference on Machine Learning*, 2022; "Probabilisitic Load Forecasting Based on Online Adaptive Learning," published in *IEEE Transactions on Power Systems*, 2021; "Minimax forward and backward learning of evolving tasks with performance guarantees," submitted to *Advances in Neural Information Processing Systems*, 2023; and "Learning in time-dependent environments based on minimax classification," to be submitted to *Journal of Machine Learning Research*. The paper "Probabilisitic Load Forecasting Based on Online Adaptive Learning" achieved the Best Applied Contribution in the Statistics Field awarded by the Spanish Society of Statistics and Operations Research-BBVA Foundation in 2022. In addition, the methods developed for minimax classification under concept drift have resulted in a patent application submitted in 2021. In the following, we discuss the four chapters in deeper detail.



(a) ESS increase of the 10-th task provided by leveraging information from preceding tasks, 13 tasks, and 20 tasks.



(b) Multidimensional adaptation to tasks' changes.

Figure 2: The proposed methodology increases the ESS leveraging information from all the tasks and provides multidimensional adaptation to task changes.

0.3.1 Learning time-dependent tasks

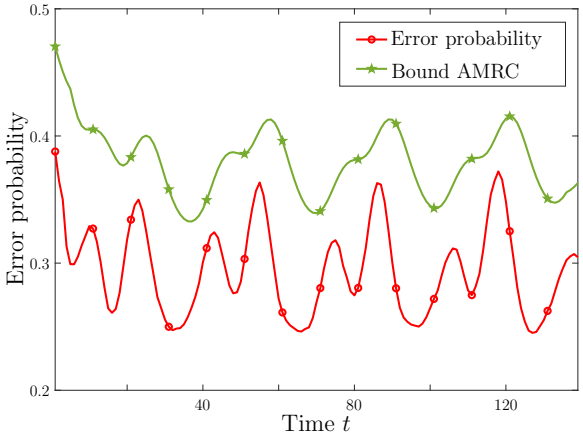
Chapter 1 establishes a learning methodology for time-dependent environments based on minimax risk classifiers (MRCs). The proposed methodology includes multiple machine learning scenarios, provides multidimensional adaptation to tasks' changes, and provides computable performance guarantees. First, we describe a general problem formulation for learning from a sequence of tasks that includes batch learning scenarios such as MDA and MTL as well as for online learning scenarios such as SCD and CL. For these settings, we develop learning techniques that account for multidimensional adaptation to time-dependent tasks by estimating multiple statistical characteristics of the varying underlying distribution. Such multidimensional adaptation can harness the change in each statistical characteristic describing the tasks. In the numerical results, we show the performance improvement of multidimensional adaptation to tasks' changes. Figure 2b shows the classification error of the proposed methodology and a state-of-the-art technique increasing the variance of the change between consecutive tasks. Such figure shows that the proposed methodology better account for multidimensional tasks' changes than state-of-the-art techniques.

The proposed methodology provides computable performance guarantees in time-dependent environments and we analytically characterize the ESS increase leveraging information from all the tasks in the sequence. Figure 2a shows the ESS increase of the 10-th task using information from the preceding tasks (tasks from 1 to 10), using information from 13 tasks (tasks from 1 to 13), and from 20 tasks (tasks from 1 to 20). Such figure shows that the ESS increases with the number of tasks, especially the ESS significantly increases when nd decreases between 1 and $1/j^2$.

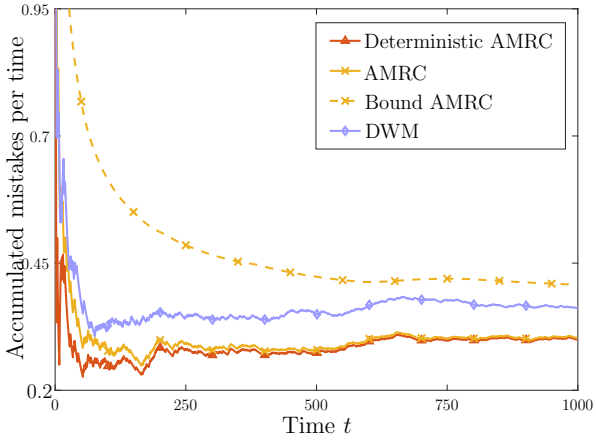
We describe in detail the efficient implementation of the proposed methodology for

Table 1: Classification error of the proposed methodology in comparison with the state-of-the-art techniques for CL.

Dataset	Yearbook		ImageNet noise		DomainNet		UTKFaces		Rotated MNIST		CLEAR	
n	10	100	10	100	10	100	10	100	10	100	10	100
GEM	43.53	23.45	39.09	13.78	69.78	53.60	12.20	12.10	45.28	32.02	56.60	8.60
MER	38.62	19.37	27.25	12.71	47.58	30.26	12.13	12.13	**36.34**	34.54	20.53	7.40
ELLA	46.36	42.98	48.75	47.11	67.15	67.35	19.10	17.79	48.13	47.96	61.15	60.43
EWC	48.94	37.17	45.75	30.68	50.03	41.17	12.13	12.13	47.05	39.93	62.73	38.53
MRCs	**14.18**	**9.42**	**14.48**	**8.68**	**33.81**	**24.53**	**10.32**	**10.32**	37.02	**21.04**	**8.22**	**4.06**



(a) Instantaneous bounds and probabilities of error for AMRC.



(b) Evolution of accumulated mistake bounds and accumulated mistakes per number of steps.

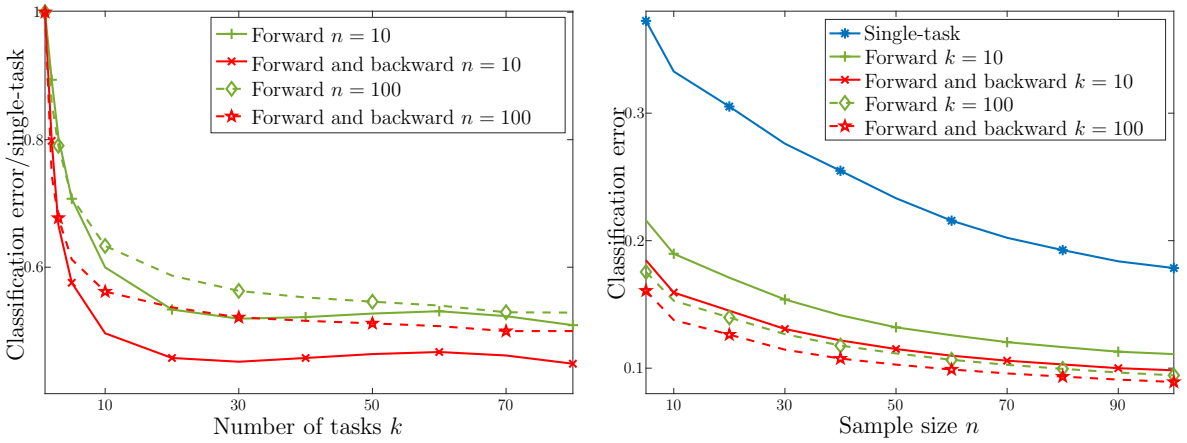
Figure 3: Results on real-world datas shows the evolution of accumulated mistake bounds.

batch learning scenarios such as MDA and MTL as well as for online learning scenarios such as SCD and CL. Using benchmark datasets, the experimental results quantify the performance improvement of the proposed methodology (MRCs) in multiple scenarios in comparison with existing techniques using $n = 10$ and $n = 100$ samples per task (see Table 1).

0.3.2 Minimax classification under concept drift

In Chapter 2, we present adaptive minimax risk classifiers (AMRCs) for supervised classification under concept drift. In the addressed setting, instance-label pairs arrive over time and each instance-label pair is sampled from a time-varying underlying distribution. Learning methods use at each time step the most recent instance-label and the classification rule at previous time to update the classification rule.

The proposed techniques account for multidimensional time changes by means of a



(a) Classification error per number of tasks.

(b) Classification error per sample size.

Figure 4: Forward and backward learning can sharply boost performance and ESS as tasks arrive.

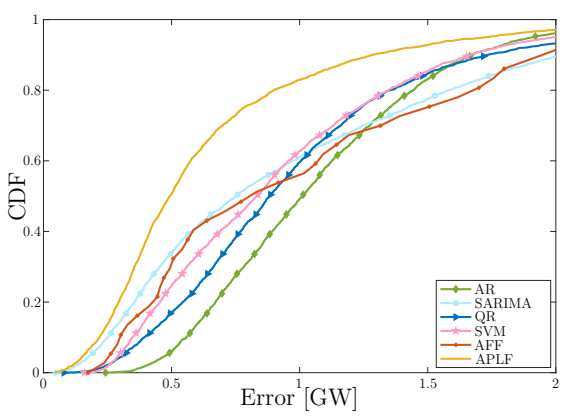
multivariate and high-order tracking of the time-varying underlying distribution. We propose tracking techniques that model the detailed evolution of the statistical characteristics of the time-varying underlying distribution using partially-observed dynamical systems. In addition, we propose techniques to learn AMRCs using an efficient subgradient method that utilizes warm-starts and maintains local approximations of the objective function. Chapter 2 describes the implementation of the proposed tracking and learning techniques.

Differently from existing techniques, AMRCs can provide computable tight performance guarantees. Specifically, the proposed techniques provide performance guarantees for AMRCs in terms of instantaneous bounds for error probabilities and bounds for accumulated mistakes. Figure 3 shows the reliability of the presented bounds. Specifically, Figure 3a shows that the instantaneous bounds can offer tight upper bounds for the error probability at each time, and Figure 3b provides comparisons of AMRC accumulated mistakes with state-of-the-art techniques and the accumulated mistake bound. As can be seen in such figure, the accumulated mistake bounds obtained at AMRC learning can offer accurate estimates for the classification error.

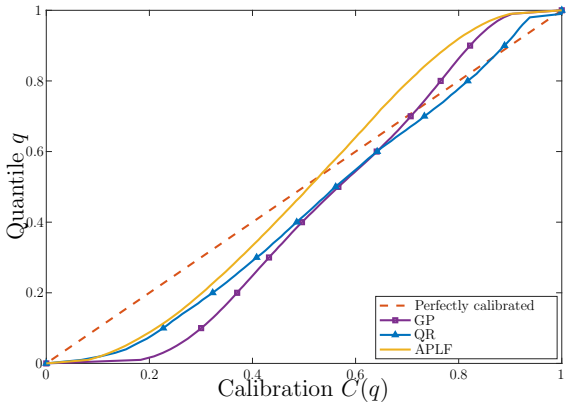
0.3.3 Minimax classification for continual learning

As a third contribution, Chapter 3 presents continual learning methods based on minimax risk classifiers (CL-MRCs). We propose learning techniques that can effectively incorporate information from the ever-increasing sequence of tasks and provide performance guarantees for forward and backward learning.

The proposed techniques effectively exploit forward and backward learning and account for time-dependent tasks. Specifically, we propose forward learning techniques that recursively use the information from preceding tasks to learn the last task in the



(a) CDFs of prediction errors.



(b) Calibration plot describes the reliability of probabilistic forecasts.

Figure 5: Performance comparison between APLF method and conventional techniques.

sequence, and we propose forward and backward learning techniques that use the information from the last task to improve performance of the sequence of tasks. Figure 4a shows the classification error of CL-MRC method divided by the classification error of single-task learning for different number of tasks with $n = 10$ and $n = 100$ sample sizes. Such figure shows that forward and backward learning can significantly improve the performance of CL-MRCs as tasks arrive. In addition, Figure 4b shows the classification error of CL-MRCs method for different sample sizes with $k = 10$ and $k = 100$ tasks. Such figures show that the methods proposed can effectively exploit backward learning that results in enhanced classification error in all the experimental results using multiple datasets, different sample sizes, and number of tasks.

The proposed techniques provide performance guarantees for forward and backward learning. To the best of our knowledge, the proposed techniques provide the first performance guarantees for CL that show positive backward transfer. In particular, the bounds for forward and backward learning are significantly lower than those for forward learning. In addition, we analytically characterize the increase in effective sample size provided by forward and backward learning in terms of the tasks' expected quadratic change and the number of tasks.

0.3.4 Adaptive probabilistic load forecasting

Chapter 4 presents techniques for adaptive probabilistic load forecasting (APLF) that can harness changes in consumption patterns and assess load uncertainties. Load forecasting is crucial for multiple energy management tasks such as scheduling generation capacity, planning supply and demand, and minimizing energy trade costs. Such relevance has increased even more in recent years due to the integration of renewable energies, electric cars, and microgrids. Conventional load forecasting techniques obtain

single-value load forecasts by exploiting consumption patterns of past load demand. However, such techniques cannot assess intrinsic uncertainties in load demand, and cannot capture dynamic changes in consumption patterns. To address these problems, Chapter 4 presents a method for probabilistic load forecasting based on the adaptive online learning of hidden Markov model (HMM). We propose learning and forecasting techniques with theoretical guarantees, and experimentally assess their performance in multiple scenarios. In particular, we develop adaptive online learning techniques that update model parameters recursively, and sequential prediction techniques that obtain probabilistic forecasts using the most recent parameters.

The performance of the method is evaluated using multiple datasets corresponding with regions that have different sizes and display assorted time-varying consumption patterns. The results show that the proposed method can significantly improve the performance of existing techniques for a wide range of scenarios. Figure 5a provides comparisons of the presented APLF with 5 representative existing techniques. Such figure shows the empirical cumulative distribution functions (CDFs) of the absolute value of prediction errors. Figure 5a shows that the proposed APLF method achieves high accuracies in comparison with existing techniques and shows that high errors occur with low probability for APLF method. For instance, the error of APLF method is less than 0.8 GW with probability 0.8, while the 5 other methods reach errors of around 1.3 GW with such probability.

Figure 5b provides quantification of the probabilistic performance of different methods. Such figure shows the correspondence between the calibration $C(q)$ of probabilistic forecasts and the quantile q. This calibration plot shows that Gaussian process (GP) and quantile regression (QR) tend to obtain forecast quantiles higher than the true quantiles, while APLF obtains more unbiased probabilistic forecasts. In particular, the true load is higher than the 50 quantile forecast load with probability very near 50% for APLF.

0.4 Background

This section briefly recalls the problem formulation and learning approaches for supervised classification and describes MRCs and HMMs.

Supervised classification Supervised classification uses training samples to determine classification rules that assign labels to instances. We denote by $\mathcal{X}$ and $\mathcal{Y}$ the sets of instances and labels, respectively; both sets are taken to be finite with $\mathcal{Y}$ represented by $\{1, 2, ..., |\mathcal{Y}|\}$. We denote by $\mathrm{T}(\mathcal{X}, \mathcal{Y})$ the set of all classification rules (both randomized and deterministic) and we denote by $\mathrm{h}(y|x)$ the probability with which rule $\mathrm{h} \in \mathrm{T}(\mathcal{X}, \mathcal{Y})$ assigns label $y \in \mathcal{Y}$ to instance $x \in \mathcal{X}$ ($\mathrm{h}(y|x) \in \{0, 1\}$ for deterministic classification rules). In addition, we denote by $\Delta(\mathcal{X} \times \mathcal{Y})$ the set of probability distributions on $\mathcal{X} \times \mathcal{Y}$ and by $\ell(\mathrm{h}, \mathrm{p})$ the expected 0-1 loss of the classification rule $\mathrm{h} \in \mathrm{T}(\mathcal{X}, \mathcal{Y})$ with respect to distribution $\mathrm{p} \in \Delta(\mathcal{X} \times \mathcal{Y})$. If $\mathrm{p}^* \in \Delta(\mathcal{X} \times \mathcal{Y})$ is the

underlying distribution of the instance-label pairs, then $\ell(\mathrm{h}, \mathrm{p}^*)$ is the error probability of rule h denoted in the following as $R(\mathrm{h})$.

Samples are usually represented as real vectors by using a feature mapping $\Phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^m$. The most common way to define such feature mapping is using multiple features over instances together with one-hot encodings of labels as follows (see e.g., [73])

$$\Phi(x, y) = \mathbf{e}_y \otimes \Psi(x) = \left[\mathbb{1}\{y = 1\}\Psi(x)^{\mathrm{T}}, \mathbb{1}\{y = 2\}\Psi(x)^{\mathrm{T}}, ..., \mathbb{1}\{y = |\mathcal{Y}|\}\Psi(x)^{\mathrm{T}}\right]^{\mathrm{T}} \quad (1)$$

where the map $\Psi : \mathcal{X} \to \mathbb{R}^d$ represents instances as real vectors. For instance, such map can be given by random Fourier features (RFF), that is

$$\Psi(x) = \left[\cos(\mathbf{u}_1^{\mathrm{T}}\mathbf{x}), \cos(\mathbf{u}_2^{\mathrm{T}}\mathbf{x}), ..., \cos(\mathbf{u}_D^{\mathrm{T}}\mathbf{x}), \sin(\mathbf{u}_1^{\mathrm{T}}\mathbf{x}), \sin(\mathbf{u}_2^{\mathrm{T}}\mathbf{x}), ..., \sin(\mathbf{u}_D^{\mathrm{T}}\mathbf{x})\right]^{\mathrm{T}}$$

for D random Gaussian vectors $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_D \sim N(0, \gamma\mathbf{I})$ with covariance given by the scaling parameter γ (see e.g., [48, 74, 75]).

Learning approaches The classification methodologies presented are based on robust risk minimization (RRM) instead of empirical risk minimization (ERM) since training samples corresponding to different tasks follow different distributions. Methods based on ERM approach for supervised classification aims to minimize the empirical expected loss of training samples, while methods based on RRM approach aims to minimize the worst-case expected loss with respect to an uncertainty set of distributions. In addition, methods based on ERM cannot provide performance guarantees in time-dependent environments since training samples follow a time-varying underlying distribution. For the methods presented, we utilize MRCs that learn classification rules by minimizing the worst-case expected loss against an uncertainty set and can provide performance guarantees. Such techniques are especially suitable for SCD because MRCs are based on expectation estimates and do not require to use training samples from the same underlying distribution.

Minimax risk classifiers MRCs learn classification rules by minimizing the worst-case error probability over distributions in an uncertainty set [76–78]. Such techniques are especially suitable for learning in time-dependent environments because MRCs determine the uncertainty sets by expectation estimates and do not require to use samples from the same underlying distribution.

MRCs learn classification rules over uncertainty sets determined by expectation estimates of a feature mapping $\Phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^m$ as

$$\mathcal{U} = \{\mathrm{p} \in \Delta(\mathcal{X} \times \mathcal{Y}) : |\mathbb{E}_{\mathrm{p}}\{\Phi(x, y)\} - \boldsymbol{\tau}| \preceq \boldsymbol{\lambda}\} \quad (2)$$

where $\boldsymbol{\tau}$ denotes the vector of expectation estimates corresponding with the feature mapping Φ and $\boldsymbol{\lambda} \succeq \mathbf{0}$ is a confidence vector that accounts for inaccuracies in the estimate.

Given the uncertainty set $\mathcal{U}$, MRC rules are solutions of the optimization problem

$$R(\mathcal{U}) = \min_{\mathrm{h}\in\mathrm{T}(\mathcal{X},\mathcal{Y})} \max_{\mathrm{p}\in\mathcal{U}} \ell(\mathrm{h},\mathrm{p}) \tag{3}$$

where $R(\mathcal{U})$ denotes the minimax risk and $\ell(\mathrm{h},\mathrm{p})$ denotes the expected loss of classification rule h for distribution p. In the following, we utilize the 0-1-loss so that the expected loss with respect to the underlying distribution is the error probability of the classification rule that is denoted by $R(\mathrm{h})$.

The MRC rule h assigns label $\hat{y} \in \mathcal{Y}$ to instance $x \in \mathcal{X}$ with probability

$$\mathrm{h}(\hat{y}|x) = \begin{cases} \left(\Phi(x,\hat{y})^{\mathrm{T}}\boldsymbol{\mu}^* - \varphi(\boldsymbol{\mu}^*)\right)_+ / c_x & \text{if } c_x \neq 0 \\ 1/|\mathcal{Y}| & \text{if } c_x = 0 \end{cases} \tag{4}$$

with

$$\varphi(\boldsymbol{\mu}^*) = \max_{x\in\mathcal{X},\mathcal{C}\subseteq\mathcal{Y}} \Big(\sum_{y\in\mathcal{C}} \Phi(x,y)^{\mathrm{T}}\boldsymbol{\mu}^* - 1 \Big) / |\mathcal{C}| \tag{5}$$

$$c_x = \sum_{y\in\mathcal{Y}} \left(\Phi(x,y)^{\mathrm{T}}\boldsymbol{\mu}^* - \varphi(\boldsymbol{\mu}^*)\right)_+ .$$

The vector parameters $\boldsymbol{\mu}^*$ is the solution of the convex optimization problem

$$\min_{\boldsymbol{\mu}} 1 - \boldsymbol{\tau}^{\mathrm{T}}\boldsymbol{\mu} + \varphi(\boldsymbol{\mu}) + \boldsymbol{\lambda}^{\mathrm{T}} |\boldsymbol{\mu}| \tag{6}$$

given by the Fenchel-Lagrange dual of (3) [76, 78]. The label that maximizes the probability in (4) is given by $\hat{y} \in \arg\max_{y\in\mathcal{Y}} \Phi(x,y)^{\mathrm{T}}\boldsymbol{\mu}^*$. The deterministic classification rule h^{d} that assigns such label $\hat{y}$ to instance x will be referred in the following as deterministic MRC.

MRCs for standard supervised classification obtain classification rules leveraging information only from a sample set $D = \{(x_i, y_i)\}_{i=1}^n$ of size n. Such methods obtain the mean and confidence vectors that determine the uncertainty set given by (2) as

$$\boldsymbol{\tau} = \frac{1}{n}\sum_{i=1}^{n} \Phi\left(x_i, y_i\right), \quad \boldsymbol{\lambda} = \lambda_0\sqrt{\boldsymbol{s}}, \quad \boldsymbol{s} = \frac{\boldsymbol{\sigma}^2}{n} \tag{7}$$

with $\boldsymbol{\sigma}^2$ an estimate of $\mathbb{V}\mathrm{ar}_{\mathrm{p}}\{\Phi(x,y)\}$, e.g., the sample variance of the n samples. The vector $\boldsymbol{s}$ describes the mean squared errors (MSEs) of the mean vector components and directly gives the confidence vector $\boldsymbol{\lambda}$ as shown in (7).

In standard supervised classification, MRCs provide bounds for the error probability $R(\mathrm{h})$ with respect to the minimax risk $R(\mathcal{U})$ and the smallest minimax risk, denoted by R^∞, as described in [76, 78]. The smallest minimax risk corresponds with uncertainty sets given by the true expectation of the feature mapping and can converge to the Bayes risk increasing the number of components of the feature mapping [78]. If the mean vector of expectation estimates is obtained as in (7), then the performance bounds of MRCs are of the usual order $\mathcal{O}(1/\sqrt{n})$ where n is the sample size. Such performance bounds of MRCs also ensure generalization for deterministic MRCs because $R(\mathrm{h}^{\mathrm{d}}) \leq 2R(\mathrm{h})$ since $1 - \mathrm{h}^{\mathrm{d}}(y|x) \leq 2(1 - \mathrm{h}(y|x))$ for any $x \in \mathcal{X}, y \in \mathcal{Y}$.

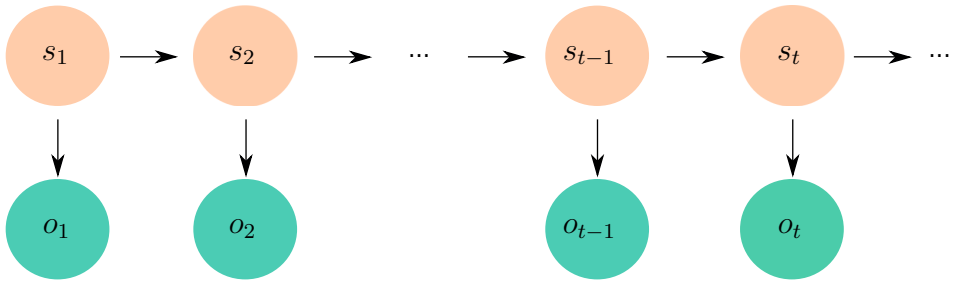


Figure 6: Hidden Markov model for sequences $\{s_t\}_{t\geq 1}$ and $\{o_t\}_{t\geq 1}$.

Hidden Markov model The methodologies presented are based on HMM also known as state-space models. Specifically, we model the relationship between states $\{s_t\}_{t\geq 1}$ and observations $\{o_t\}_{t\geq 1}$ as a HMM. Such models allow to predict hidden states from past states and observations, and are determined by conditional distributions that represents the relationship between two following states and between each state and observations (see Figure 6). We model the sequence of states and observations as a non-homogeneous HMM so that both conditional distributions change in time. Such dynamic modelling allows to adapt to changes in the underlying distribution. In the classification methodologies proposed, the states are expectations of each task and the observations are sample averages; while in the regression method, the states are load demands at each time step an the observations are variables related to load demand such as weather factors.

Chapter 1

Learning Time-dependent Tasks

1.1 Introduction

Learning a sequence of classification problems (tasks) is often of interest in multiple machine learning scenarios including MDA (e.g., [1,2]), MTL (e.g., [3,4]), SCD (e.g., [5,6]), and CL (e.g., [7, 8]). In such a sequence, it is common that tasks are time-dependent in the sense that consecutive tasks often have a higher similarity. An example of such evolving tasks is the classification of portraits from different time periods [19]; in that case, the similarity between consecutive tasks (portraits of consecutive time periods) is significantly higher. Learning a sequence of tasks holds promise to significantly improve performance by leveraging information from different tasks. Such transfer of information can enable accurate classification even in cases with reduced sample sizes, thus significantly increasing the ESS of each task. However, exploiting the benefits of transferring information is challenging since tasks are characterized by different underlying distributions [12–16].

This chapter presents a learning methodology for time-dependent environments that includes multiple machine learning scenarios, account for multidimensional tasks' changes, and provides computable performance guarantees. Specifically, the main contributions of this chapter are as follows.

- We establish a learning methodology for classification in time-dependent environments based on MRCs. Such methodology includes multiple scenarios such as MDA, SCD, MTL, and CL.
- We develop learning techniques that account for multidimensional adaptation to time-dependent tasks by estimating multiple statistical characteristics of the varying underlying distribution.
- We show that the proposed methodology can provide computable tight performance guarantees in time-dependent environments and also show the ESS increase of each task using information from other tasks.

- Using benchmark datasets, numerical results quantify the performance improvement of the proposed methodology in multiple scenarios and the reliability of the performance guarantees.

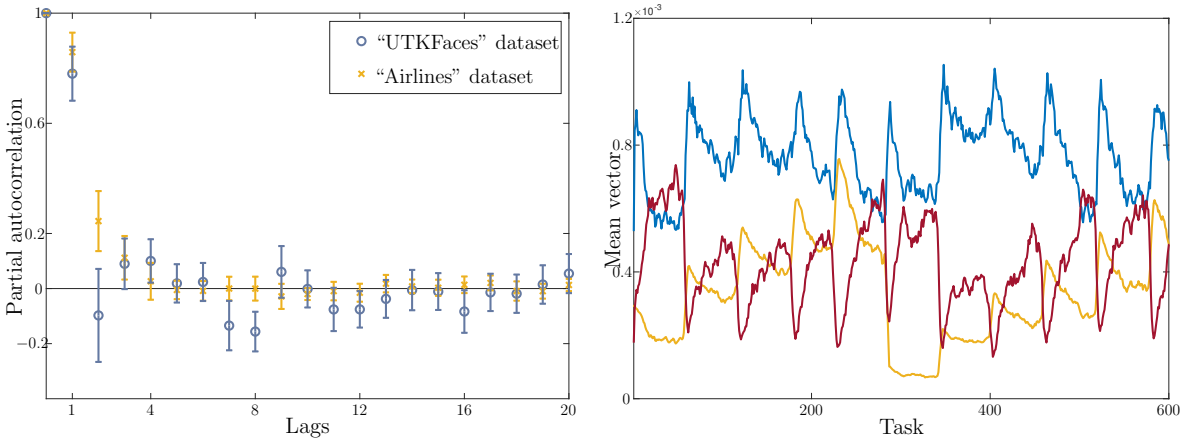
The rest of this chapter is organized as follows. Section 1.2 briefly describes the problem formulation and MRCs. Section 1.3 presents the learning methodology for classification in time-dependent environments and describes the performance guarantees. Sections 1.4 and 1.5 propose sequential techniques for efficient learning the proposed methods and analytically characterize the ESS increase, respectively. We describe the implementation in scenarios under time-dependent environments in Section 1.6. Then, Section 1.7 includes numerical results.

Notation Calligraphic letters represent sets; bold lowercase letters represent vectors; bold uppercase letters represent matrices; $\mathbf{I}$ and $\mathbb{I}\{\cdot\}$ denote the identity matrix and the indicator function, respectively; $(\cdot)_+$ denotes the positive part of its argument; $\operatorname{sign}(\cdot)$ denotes the vector given by the signs of the argument components; $\mathbf{e}_i$ denotes the i-th vector in a standard basis, $\|\cdot\|_1$ and $\|\cdot\|_\infty$ denote the 1-norm and the infinity norm of its argument, respectively; $\preceq$ and $\succeq$ denote vector inequalities; and $\mathbb{E}_{\mathrm{p}}\{\cdot\}$ and $\mathbb{V}\mathrm{ar}_{\mathrm{p}}\{\cdot\}$ denote the expectation and the variance of its argument with respect to distribution p. For a vector $\mathbf{v}$, $v^{(i)}$ and $\mathbf{v}^{\mathrm{T}}$ denote the i-th component and the transpose of $\mathbf{v}$, respectively. In addition, non-linear operators acting on vectors of the same dimension denote component-wise operations. For instance, $|\mathbf{v}|$ and $\mathbf{v}^2$ denote the vector formed by the absolute value and the square of each component, respectively.

1.2 Problem Formulation

This section describes the problem formulation.

In supervised classification from a sequence of tasks, sample sets $D_1, D_2, \ldots, D_k$ correspond with different classification tasks characterized by underlying distributions $\mathrm{p}_1, \mathrm{p}_2, \ldots, \mathrm{p}_k$. Learning methods aim to obtain classification rules $\{\mathrm{h}_j\}_{i\leq j\leq k}$ for a sequence of tasks with small expected losses $\{\ell(\mathrm{h}_j, \mathrm{p}_j)\}_{i\leq j\leq k}$ for $1 \leq i \leq k$. Such settings are common in machine learning paradigms including batch learning scenarios such as MDA and MTL and online learning scenarios such as SCD and CL. MDA uses j sample sets $D_1, D_2, \ldots, D_j$ corresponding with different classification tasks (source domains) to obtain a classification rule for the k-th task (target domain); while MTL uses k sample sets $D_1, D_2, \ldots, D_k$ corresponding with different tasks to obtain classification rules for the k tasks in the sequence. In SCD, sample sets $D_1, D_2, \ldots$ corresponding with different tasks arrive over time and SCD uses, at each step k, sample set D_{k-1} and the classification rule at previous time to obtain a classification rule for the k-th task; while in CL, sample sets $D_1, D_2, \ldots$ corresponding with different tasks arrive over time and CL uses, at each step k, sample set D_k and information acquired from sample sets $D_1, D_2, \ldots, D_{k-1}$ to obtain classification rules for the k tasks in the sequence.



(a) Averaged partial autocorrelation of mean vectors components +/- their standard deviations.

(b) Mean vector components using "Airlines" dataset.

Figure 1.1: Multidimensional time-dependence between consecutive tasks.

Most existing techniques for learning from a sequence of tasks are designed for tasks sampled i.i.d. from a task environment, that is the tasks' distributions

$$\mathrm{p}_j \text{ are independent and identically distributed for } j = 1, 2, \ldots. \qquad \text{(i.i.d.-A)}$$

In the following, we propose techniques designed for time-dependent tasks, that is consecutive tasks are significantly more similar. Specifically, we assume that the changes between consecutive tasks satisfy

$$\mathrm{p}_j - \mathrm{p}_{j-1} \text{ are independent and zero-mean for } j = 2, 3, \ldots. \qquad \text{(TD-A)}$$

Section 1.3 below assesses that the TD-A assumption can better account for the usual higher similarities between consecutive tasks than the i.i.d.-A assumption using real-world datasets.

1.3 Learning in time-dependent environments

This section assesses the time-dependent assumption in Section 1.2, presents the methodology for learning in time-dependent environments, and describes the performance guarantees of the proposed methodology.

1.3.1 Time-dependent tasks

This section describes the main aspects of time-dependent tasks and the inadequacy of conventional methods in time-dependent environments. In particular, we analyze the similarities among underlying distributions of time-dependent tasks p_j, for $j = 1, 2, \ldots$,

by assessing the similarities among the corresponding vectors formed by the expectations of a feature mapping. For each j-th task, such mean vector, denoted by $\boldsymbol{\tau}_j^\infty = \mathbb{E}_{\mathrm{p}_j}\{\Phi(x,y)\}$, represents the statistical characteristics of the underlying distribution p_j as measured by the feature mapping $\Phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^m$.

For a sequence of time-dependent tasks, the similarities between two tasks depend on their distance in the sequence and the similarities between p_{j+t} and p_j are often higher for small t. In particular, consecutive tasks are significantly more similar. Figure 1.1a shows the averaged partial autocorrelation of the mean vector components +/- their standard deviations for different lags using "Airlines" and "UTKFaces" datasets (see dataset characteristics in Table 1.1 of Section 1.7). "Airlines" dataset is divided in segments of samples corresponding to consecutive times where each task corresponds to each of those segments, and "UTKFaces" dataset is composed by tasks corresponding to the classification of face images of a specific age. Figure 1.1a displays that the correlation of the sequence of mean vectors decreases as the lags increase with a highest value for lag 1, that is, the correlation between mean vectors $\boldsymbol{\tau}_{j+t}^\infty$ and $\boldsymbol{\tau}_j^\infty$ decreases with t and is significantly higher for consecutive tasks ($t = 1$).

The TD-A assumption in Section 1.2 can better describe similarities among tasks in time-dependent environments than the conventional i.i.d.-A assumption. For instance, for any j-th task, the TD-A assumption implies that $\mathrm{p}_{j+t} - \mathrm{p}_j$ is a zero-mean random variable with variance that decreases with t since $\mathbb{V}\mathrm{ar}\{\mathrm{p}_{j+t} - \mathrm{p}_j\} = \sum_{i=1}^{t} \mathbb{V}\mathrm{ar}\{\mathrm{p}_{j+i} - \mathrm{p}_{j+i-1}\}$, while i.i.d.-A would imply that $\mathrm{p}_{j+t} - \mathrm{p}_j$ is a zero-mean random variable with variance that does not depend on t since $\mathbb{V}\mathrm{ar}\{\mathrm{p}_{j+t} - \mathrm{p}_j\} = \mathbb{V}\mathrm{ar}\{\mathrm{p}_{j+1} - \mathrm{p}_j\} = 2\mathbb{V}\mathrm{ar}\{\mathrm{p}_1\}$ for any t and j. The better adequacy of the TD-A assumption for time-dependent tasks is also reflected in the partial autocorrelations of Figure 1.1a. Note that if the tasks' distributions satisfy i.i.d.-A, then the partial autocorrelations of the mean vector components would be zero at any lag; while if the tasks' distributions satisfy TD-A, then the partial autocorrelations of the mean vector components are larger than zero at lag 1. Differently from i.i.d.-A, TD-A can capture the usual higher similarities between consecutive tasks.

The similarities among tasks are often multidimensional and different statistical characteristics of the underlying distribution change in a different manner. Figure 1.1b shows mean vector components for different tasks using "Airlines" dataset. Such figure illustrates a clearly different change in each mean vector component that reflects multidimensional tasks' changes. Existing methods for time-dependent tasks account for a scalar rate of change that cannot capture the multidimensional tasks' changes. In the following, we propose techniques that account for the change in each component in the mean vector by using a vector $\boldsymbol{d}_j$ that assesses the expected quadratic change $\mathbb{E}\{\boldsymbol{w}_j^2\} = \mathbb{E}\{(\boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_{j-1}^\infty)^2\}$ between each component in the mean vector.

1.3.2 Learning methodology

Figure 1.2 depicts the flow diagram for the proposed methodology for learning in time-dependent environments that obtains MRC rules for each task. The proposed method-

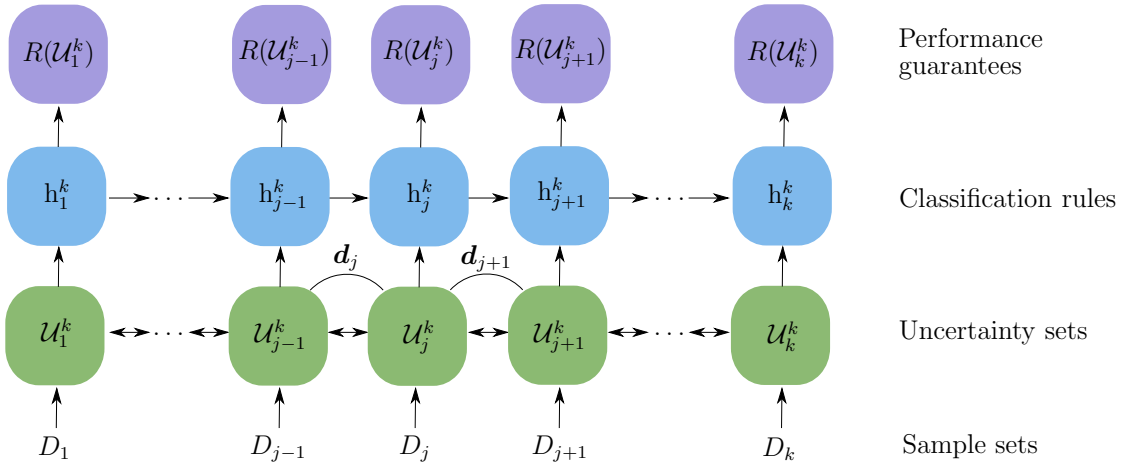


Figure 1.2: Diagram for the proposed methodology.

ology provides multidimensional adaptation to tasks' changes, provides computable performance guarantees, and includes multiple machine learning scenarios.

The learning methodology proposed obtains, for each j-th task, an uncertainty set $\mathcal{U}_j^k$ leveraging information from k tasks and accounting for the expected quadratic change between consecutive tasks. Specifically, each uncertainty set $\mathcal{U}_j^k$ is determined as in (2) by mean and confidence vectors that are obtained using those vectors for consecutive tasks and the expected quadratic change between consecutive mean vectors $\boldsymbol{d}_j$ and $\boldsymbol{d}_{j+1}$. Section 1.4 below describes techniques that recursively obtain mean and MSE vectors for each task. Then, we obtain for each j-th task the classification rule h_j^k and the minimax risk $R(\mathcal{U}_j^k)$ using the uncertainty set $\mathcal{U}_j^k$. Each classification rule h_j^k and the minimax risk $R(\mathcal{U}_j^k)$ are determined by classifier parameters that are obtained solving the convex optimization problem (6). Section 2.4 describes techniques to solve the convex optimization problem in (6) based on conventional methods [79, 80].

The learning methodology adapts to time-dependent tasks by accounting for the change between consecutive tasks, and accounts for multidimensional changes by estimating over time multiple statistical characteristics of the underlying distribution. Section 1.4 below describes techniques that estimate the evolution over tasks of each mean vector component accounting for the expected quadratic change between consecutive mean vector components.

The proposed learning methodology provides computable tight performance guarantees by assessing the minimax risk. Section 1.3.3 shows computable tight performance guarantees for error probabilities of each task in terms of the minimax risks $R(\mathcal{U}_j^k)$ obtained at learning. In addition, the proposed techniques boost the ESS of each task by effectively leveraging information from other tasks in the sequence. Section 1.5 below analytically characterizes the ESS increase provided by the proposed methodology in terms of the expected quadratic change and the number of tasks.

The proposed methodology can be used in multiple machine learning paradigms.

Section 1.6 below shows the implementation of the proposed methodology in batch learning scenarios including MDA and MTL and online learning scenarios including SCD and CL. The proposed methodology enables to obtain classification rules in online and batch learning scenarios and effectively use information from all the tasks in the sequence to obtain a classification rule or a sequence of classification rules.

1.3.3 Performance guarantees

The proposed methodology provides computable tight performance guarantees for the error probability of each j-th task leveraging information from k tasks. Let $R(\mathrm{h}_j^k)$ denote the error probability of the classification rule h_j^k determined by the parameter $\boldsymbol{\mu}_j^k$. Then, using the bounds of [78] in the addressed setting, we have that

$$R(\mathrm{h}_j^k) \leq R(\mathcal{U}_j^k) + \left(\left|\boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_j^k\right| - \boldsymbol{\lambda}_j^k\right)^{\mathrm{T}} \left|\boldsymbol{\mu}_j^k\right| \tag{1.1}$$

where $R(\mathcal{U}_j^k)$ and $\mathcal{U}_j^k$ denote the minimax risk and the uncertainty set given as in (2) by mean and confidence vectors $\boldsymbol{\tau}_j^k$ and $\boldsymbol{\lambda}_j^k$.

The above inequality provides computable tight bounds for error probabilities given by the minimax risk. Specifically, the minimax risk $R(\mathcal{U}_j^k)$ directly provides a bound if the error in the mean vector estimate is not underestimated, i.e., $\boldsymbol{\lambda}_j^k \succeq |\boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_j^k|$. In other cases, the minimax risk $R(\mathcal{U}_j^k)$ still provides approximate bounds as long as the underestimation of the mean vector estimate is not substantial. Section 1.7 shows that the presented bounds can provide tight performance guarantees for the proposed methodology in practice.

Computable tight performance guarantees are essential to estimate the classification error of tasks with limited sample sizes because in these cases the classification error cannot be reliably estimated using cross-validation methods [81, 82]. As described in Section 1.1, most conventional techniques for learning from a sequence of tasks do not provide performance guarantees since tasks are characterized by different underlying distributions. Certain techniques designed for time-dependent tasks provide performance guarantees but cannot be computed in practice [64, 71, 72]; while existing techniques that provide computable performance guarantees are not designed for time-dependent tasks [68–70]. The proposed methodology provides computable tight bounds for error probabilities in time-dependent environments that allow to assess the performance of each task.

Certain techniques for SCD and [64] for CL can provide performance guarantees in time-dependent environments for error probabilities of each task with respect to empirical risks (generalization bounds) [64, 71, 72]. Such bounds are non-computable in practice because are given in terms of the discrepancy and the Kullback-Leibler divergence between consecutive distributions which are not possible to accurately estimate from data. Certain techniques for MDA can provide performance guarantees [1,65–67] but are not designed for time-dependent tasks and are non-computable in practice and because are given in terms of distribution discrepancies between the source tasks (domains) and

the target task. Certain techniques for MTL and CL can provide computable performance guarantees with respect to empirical multitask error based on PAC-Bayes theory (PAC-Bayes generalization bounds) [68–70,83,84]. Such techniques provide bounds that can be computed in practice but are not designed for time-dependent tasks.

1.4 Determining uncertainty sets from sample sets in time-dependent environments

This section presents techniques to obtain for each task an uncertainty set accounting for the evolution of the tasks' distribution and leveraging information from all tasks.

The proposed techniques recursively obtain uncertainty sets for each task leveraging information from all the tasks in the sequence and accounting for time-dependent tasks. Specifically, we recursively obtain mean and MSE vectors using those for consecutive tasks and the expected quadratic change between consecutive tasks. Once mean and MSE vectors are obtained, we take the confidence vector of each task from the MSE vector as in (7).

For a sequence of k tasks, the proposed techniques first obtain for each j-th task with $j = \{1, 2, \ldots, k\}$ the mean vector $\boldsymbol{\tau}_j^j$ together with the MSE vector $\boldsymbol{s}_j^j$ that leverage information up to the j-th task using those for the previous task $\boldsymbol{\tau}_{j-1}^{j-1}, \boldsymbol{s}_{j-1}^{j-1}$ as

$$\boldsymbol{\tau}_j^j = \boldsymbol{\tau}_{j-1}^{j-1} + \boldsymbol{\eta}_j^j \left(\boldsymbol{\tau}_j - \boldsymbol{\tau}_{j-1}^{j-1}\right) \tag{1.2}$$

$$\boldsymbol{s}_j^j = \boldsymbol{\eta}_j^j \boldsymbol{s}_j \tag{1.3}$$

$$\boldsymbol{\eta}_j^j = \frac{\boldsymbol{s}_{j-1}^{j-1} + \boldsymbol{d}_j}{\boldsymbol{s}_j + \boldsymbol{s}_{j-1}^{j-1} + \boldsymbol{d}_j} \tag{1.4}$$

for $j = 2, 3, \ldots, k$ with $\boldsymbol{\tau}_j, \boldsymbol{s}_j$ given by (7), $\boldsymbol{\tau}_1^1 = \boldsymbol{\tau}_1$, and $\boldsymbol{s}_1^1 = \boldsymbol{s}_1$. Mean and MSE vectors $\boldsymbol{\tau}_j^j, \boldsymbol{s}_j^j$ are obtained leveraging information from sample sets $D_1, D_2, \ldots, D_j$. Specifically, for each j-th task, those vectors are obtained by acquiring information from sample set D_j through mean and MSE vectors $\boldsymbol{\tau}_j, \boldsymbol{s}_j$ and retaining information from sample sets $D_1, D_2, \ldots, D_{j-1}$ through mean and MSE vectors $\boldsymbol{\tau}_{j-1}^{j-1}, \boldsymbol{s}_{j-1}^{j-1}$. The proposed methodology allows to obtain mean and MSE for the j-th task with samples up to the $(j-1)$-th task (i.e., $|D_j| = 0$) from those for the previous task $\boldsymbol{\tau}_{j-1}^{j-2}, \boldsymbol{s}_{j-1}^{j-2}$ as

$$\boldsymbol{\tau}_j^{j-1} = \boldsymbol{\tau}_{j-1}^{j-2} + \boldsymbol{\eta}_j^{j-1} \left(\boldsymbol{\tau}_{j-1} - \boldsymbol{\tau}_{j-1}^{j-2}\right) \tag{1.5}$$

$$\boldsymbol{s}_j^{j-1} = \boldsymbol{\eta}_j^{j-1} \boldsymbol{s}_{j-1} + \boldsymbol{d}_j \tag{1.6}$$

$$\boldsymbol{\eta}_j^{j-1} = \frac{\boldsymbol{s}_{j-1}^{j-2}}{\boldsymbol{s}_{j-1} + \boldsymbol{s}_{j-1}^{j-2}} \tag{1.7}$$

for $j = 3, 4, \ldots k$ with $\boldsymbol{\tau}_j, \boldsymbol{s}_j$ given by (7), $\boldsymbol{\tau}_2^1 = \boldsymbol{\tau}_1$, and $\boldsymbol{s}_2^1 = \boldsymbol{d}_2$. Mean and MSE vectors $\boldsymbol{\tau}_j^{j-1}, \boldsymbol{s}_j^{j-1}$ are obtained leveraging information from sample sets $D_1, D_2, \ldots, D_{j-1}$.

Specifically, for each j-th task, those vectors are obtained by acquiring information from sample set D_{j-1} through mean and MSE vectors $\boldsymbol{\tau}_{j-1}, \boldsymbol{s}_{j-1}$ and retaining information from sample sets $D_1, D_2, \ldots, D_{j-2}$ through mean and MSE vectors $\boldsymbol{\tau}_{j-1}^{j-2}, \boldsymbol{s}_{j-1}^{j-2}$. The above recursions include settings where information from each j-th tasks arrives after obtaining mean and MSE vectors $\boldsymbol{\tau}_j^{j-1}, \boldsymbol{s}_j^{j-1}$ as in SCD. Such mean and MSE vectors can also be obtained from mean and MSE vectors $\boldsymbol{\tau}_{j-1}^{j-1}, \boldsymbol{s}_{j-1}^{j-1}$ given by (1.2), (1.3) as $\boldsymbol{\tau}_j^{j-1} = \boldsymbol{\tau}_{j-1}^{j-1}, \boldsymbol{s}_j^{j-1} = \boldsymbol{s}_{j-1}^{j-1} + \boldsymbol{d}_j$.

The proposed techniques obtain for each j-th task the mean vector estimate $\boldsymbol{\tau}_j^k$ together with the MSE vector $\boldsymbol{s}_j^k$ that leverage information from the k tasks as

$$\boldsymbol{\tau}_j^k = \boldsymbol{\tau}_{j+1}^k + \boldsymbol{\eta}_j^k \left(\boldsymbol{\tau}_j^j - \boldsymbol{\tau}_{j+1}^k\right) \tag{1.8}$$

$$\boldsymbol{s}_j^k = \boldsymbol{s}_{j+1}^k + \boldsymbol{\eta}_j^k (\boldsymbol{s}_j^j - 2\boldsymbol{s}_{j+1}^k + \boldsymbol{\eta}_j^k \boldsymbol{s}_{j+1}^k) \tag{1.9}$$

$$\boldsymbol{\eta}_j^k = \frac{\boldsymbol{d}_{j+1}}{\boldsymbol{s}_j^j + \boldsymbol{d}_{j+1}} \tag{1.10}$$

for $j = 1, 2, \ldots, k-1$ with $\boldsymbol{\tau}_j^j, \boldsymbol{s}_j^j$ given by (1.2)-(1.3). Mean and MSE vectors $\boldsymbol{\tau}_j^k$ and $\boldsymbol{s}_j^k$ are obtained leveraging information from sample sets $D_1, D_2, \ldots, D_k$. Specifically, for each j-th task, those vectors are obtained acquiring information from sample sets $D_{j+1}, D_{j+2}, \ldots, D_k$ through mean and MSE vectors $\boldsymbol{\tau}_{j+1}^k, \boldsymbol{s}_{j+1}^k$ and retaining information from sample sets $D_1, D_2, \ldots, D_j$ through mean and MSE vectors $\boldsymbol{\tau}_j^j, \boldsymbol{s}_j^j$.

The above recursions adapt to multidimensional tasks' changes by accounting for the change between consecutive mean vector components. Specifically, for $i = 1, 2, \ldots, m$, the i-th component of the mean vector estimate is updated by using the corresponding component of the expected quadratic change, the estimate for the consecutive task, and the most recent sample average. Such update accounts for the specific evolution of each i-th component of the mean vector through the recursions in equations (1.5), (1.2), and (1.8) and gain vectors in equations (1.7), (1.4), and (1.10). In particular, updates for mean vector components with a low gain slightly change the estimate for the consecutive task (previous task in (1.2), (1.5) and next task in (1.8)), while those updates for components and tasks with a high gain increase the relevance of the information of the corresponding task (sample average in (1.2), (1.5) and mean vector estimate in (1.8)).

The result below shows that the mean vectors obtained above are the estimates of the mean vector $\boldsymbol{\tau}_j^\infty$ for any j with minimum MSE.

Theorem 1. Let $\boldsymbol{\sigma}_j^2$ and $\boldsymbol{d}_j$ in recursions (1.2)-(1.10) be such that $\boldsymbol{\sigma}_j^2 = \mathbb{V}\mathrm{ar}_{\mathrm{p}_j}\{\Phi(x, y)\}$ and $\boldsymbol{d}_j = \mathbb{E}\{\boldsymbol{w}_j^2\} = \mathbb{E}\{(\boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_{j-1}^\infty)^2\}$. If the tasks' distributions satisfy (TD-A), then we have that

1. $\boldsymbol{\tau}_j^j$ given by (1.2) is the unbiased linear estimator of the mean vector $\boldsymbol{\tau}_j^\infty$ based on $D_1, D_2, \ldots, D_j$ that has the minimum MSE and $\boldsymbol{s}_j^j$ given by (1.3) is its MSE. In particular, if $|D_j| = 0$, then $\boldsymbol{\tau}_j^{j-1}$ given by (1.5) is the unbiased linear estimator of the mean vector $\boldsymbol{\tau}_j^\infty$ based on $D_1, D_2, \ldots, D_{j-1}$ that has the minimum MSE and $\boldsymbol{s}_j^{j-1}$ given by (1.6) is its MSE.

2. $\boldsymbol{\tau}_j^k$ given by (1.8) is the unbiased linear estimator of the mean vector $\boldsymbol{\tau}_j^\infty$ based on $D_1, D_2, \ldots, D_j, \ldots, D_k$ that has the minimum MSE and and $\boldsymbol{s}_j^k$ given by (1.9) is its MSE.

Proof. See Appendix A.1. □

The above theorem shows that equations in (1.2)-(1.10) enable to recursively obtain, for each j-th task, mean vector estimate as well as its MSE vectors leveraging all the information from the k tasks in the sequence.

The vector $\boldsymbol{d}_j$ assesses the expected quadratic change between consecutive tasks $\mathbb{E}\{\boldsymbol{w}_j^2\} = \mathbb{E}\{(\boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_{j-1}^\infty)^2\}$. Such expectation estimate can be obtained as the sample average of the most recent samples as

$$\boldsymbol{d}_j = \frac{1}{W} \sum_{l=1}^{W} \left(\boldsymbol{\tau}_{j_l} - \boldsymbol{\tau}_{j_{l-1}}\right)^2 \tag{1.11}$$

where $j_0, j_1, \ldots, j_W$ are the $W+1$ closest indexes to j and sample average $\boldsymbol{\tau}_j$ is given by (7). In addition, vector $\boldsymbol{\sigma}_j^2$ can be estimated as the sample variance of D_j.

1.5 Effective sample sizes

This section analytically characterizes the ESS increase leveraging information from all the tasks in the sequence. The ESS commonly quantifies the performance improvement of an algorithm in terms of the number of samples the baseline method would require to achieve the same performance. The ESS shows the improvement of leveraging information from other tasks in the sequence in comparison with learning each task as in standard supervised classification (single task learning).

In standard supervised classification, MRCs provide bounds for the minimax risk for each j-th task with respect to the smallest minimax corresponding with the optimal minimax rules. Such rules correspond with uncertainty sets given by the true expectation of the feature mapping Φ, that is

$$\mathcal{U}_j^\infty = \{\mathrm{p} \in \Delta(\mathcal{X} \times \mathcal{Y}) : \mathbb{E}_\mathrm{p}\{\Phi(x, y)\} = \boldsymbol{\tau}_t\}.$$

The minimum worst-case error probability over distributions in $\mathcal{U}_j^\infty$ is given by

$$R_j^\infty = \min_{\boldsymbol{\mu}} 1 - \boldsymbol{\tau}_j^{\infty\mathrm{T}} \boldsymbol{\mu} + \varphi(\boldsymbol{\mu}) = 1 - \boldsymbol{\tau}_j^{\infty\mathrm{T}} \boldsymbol{\mu}_j^\infty + \varphi(\boldsymbol{\mu}_j^\infty)$$

corresponding with the rule given by parameters $\boldsymbol{\mu}_j^\infty$. Such classification rule is referred to as optimal minimax rule becase for any uncertainty set $\mathcal{U}_j^\infty$ given by (2) that contains the underlying distribution, we have that $\mathcal{U}_j^\infty \subseteq \mathcal{U}_j^k$ and hence $R_j^\infty \leq R(\mathcal{U}_j)$. This optimal minimax rule could only be obtained by an exact estimation of the mean vector that in turn would require an infinite amount of samples per task and can converge to the Bayes risk increasing the number of components of the feature mapping [78].

The following result provides bounds for the minimax risk for each j-th task with respect to the smallest minimax risk using mean and MSE vectors $\boldsymbol{\tau}_j^j, \boldsymbol{s}_j^j$ obtained as in (1.2) and (1.3).

Theorem 2. Let M and κ be such that $M \geq \|\Phi(x,y)\|_\infty \; \forall \; (x,y) \in \mathcal{X} \times \mathcal{Y}$ and

$$\kappa \geq \max_{\substack{i=1,2,\ldots,m,\\ j=1,2,\ldots,k}} \left\{ \frac{\sigma\left(\Phi_j^{(i)}\right)}{\sigma_j^{(i)}}, \frac{\sigma\left(w_j^{(i)}\right)}{\sqrt{d_j^{(i)}}} \right\}$$

where $\Phi_j^{(i)}$ denotes the r.v. given by the i-th component of the feature mapping for samples from the j-th task, $\sigma(z)$ denotes the sub-Gaussian parameter of a r.v. z, i.e., $\mathbb{E}\{e^{t(z-\mathbb{E}\{z\})}\} \leq e^{\sigma(z)^2 t^2/2} \; \forall t$. Then, with probability at least $1-\delta$, we have that

$$R(\mathcal{U}_j^j) \leq R_j^\infty + \frac{M(\kappa+1)\sqrt{2\log(2m/\delta)}}{\sqrt{n_j^j}} \left\|\boldsymbol{\mu}_j^\infty\right\|_1 \tag{1.12}$$

with

$$n_j^j \geq n_j + n_{j-1}^{j-1} \frac{\|\boldsymbol{\sigma}_j^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty + \|\boldsymbol{d}_j\|_\infty n_{j-1}^{j-1}} \tag{1.13}$$

for $j = 2, 3, \ldots k$ and $n_1^1 = n_1$.

Proof. See Appendix A.2. □

The excess risk in inequality (1.12) decreases as $\mathcal{O}(\sqrt{n_j^j})$, while such difference would decrease as $\mathcal{O}(\sqrt{n_j})$ using only the information from the j-th task as in standard supervised classification. Therefore, n_j^j in (1.12) is the ESS of the proposed methodology leveraging information from j tasks. The ESS of each task n_j^j is given by the ESS for the preceding task n_{j-1}^{j-1}, the sample size n_{j-1}, and the expected quadratic change $\boldsymbol{d}_j$. In particular, if the expected quadratic change $\boldsymbol{d}_j$ is small, the ESS is given by the sample size, while if $\boldsymbol{d}_j$ is small, the ESS is given by the sum of the sample size and the ESS of the previous task.

The ESS in Theorem 2 above increases with the number of tasks and decreases with the expected quadratic change $\boldsymbol{d}_j$ between consecutive distributions. The coefficient κ in (1.12) can be taken to be small as long as the values used for σ_j and $\sqrt{d_j}$ are not much lower than the sub-Gaussian parameters of Φ_j and w_j, respectively. In particular, κ is smaller than the maximum of $M/\min_{j,i}\{\sigma_j^{(i)}\}$ and $2M/\min_{j,i}\{\sqrt{d_j^{(i)}}\}$ with $M \geq \|\Phi(x,y)\|_\infty$ for any $x \in \mathcal{X}, y \in \mathcal{Y}$ due to the bound for the sub-Gaussian parameter of bounded random variables (see e.g., Section 2.1.2 in [85]).

Theorem 2 shows the ESS in terms of the ESS of the previous task. The following result allows to directly quantify the ESS in terms of the sample size and the expected quadratic change.

Theorem 3. Let d, $\boldsymbol{\sigma}_j$, and n be such that $d \geq \|\boldsymbol{d}_j\|_\infty$, $\|\boldsymbol{\sigma}_j^2\|_\infty \leq 1$, and $n \leq n_j$ for $j = 1, 2, \ldots, k$. Then, we have that the ESS in (1.12) satisfies

$$n_j^j \geq n\left(1 + \frac{(1+\alpha)^{2j-1} - 1 - \alpha}{\alpha(1+\alpha)^{2j-1} + \alpha}\right) \quad \text{with} \quad \alpha = \frac{2}{\sqrt{1 + \frac{4}{nd}} - 1}. \tag{1.14}$$

In particular, if $nd < \frac{1}{j^2}$, then we have that $n_j^j \geq n\left(1 + \frac{j-1}{3}\right)$ for $j > 1$.

Proof. See Appendix A.3. □

The above theorem characterizes the increase in ESS for each j-th task provided by the proposed methodology leveraging information from the $j-1$ previous tasks in terms of the tasks' expected quadratic change. Such increase grows monotonically with the number of tasks $j-1$ as shown in (1.14) and becomes proportional to $j-1$ when the expected quadratic change is smaller than $1/(j^2 n)$.

Analogously to bounds in Theorems 2 and 3, the proposed methodology also increases the ESS of each task using mean and MSE vectors $\boldsymbol{\tau}_j^{j-1}, \boldsymbol{s}_j^{j-1}$ obtained as in (1.5) and (1.6). The proposed methodology satisfies inequality (1.12) with $R(\mathcal{U}_j^{j-1})$ and n_j^{j-1} instead of $R(\mathcal{U}_j^j)$ and n_j^j with

$$n_j^{j-1} \geq (n_{j-1} + n_{j-1}^{j-2}) \frac{\|\boldsymbol{\sigma}_{j-1}^2\|_\infty}{\|\boldsymbol{\sigma}_{j-1}^2\|_\infty + \|\boldsymbol{d}_j\|_\infty (n_{j-1} + n_{j-1}^{j-2})}$$

for $j = 2, 3, \ldots, k$ and $n_1^0 = 0$. Taking d, $\boldsymbol{\sigma}_j$, and n as in Theorem 3, the ESS above satisfies

$$n_j^{j-1} \geq n \frac{(1+\alpha)^{2j-1} - 1 - \alpha}{\alpha(1+\alpha)^{2j-1} + \alpha} \quad \text{with} \quad \alpha = \frac{2}{\sqrt{1 + \frac{4}{nd}} - 1} \tag{1.15}$$

for any $j = 1, 2, \ldots, k$ (see Appendices A.4 and A.5).

Learning techniques commonly adapt to time-dependent environments using sliding windows of previous tasks [6, 57, 58, 62, 86, 87]. Large window size values adapt to gradual changes in the tasks' distributions, while small window size values adapt to abrupt changes in the tasks' distributions. Techniques based on sliding windows obtain classification rules for each task using the $\bar{W}$ most recent sample sets with $\bar{W}$ the window size value. If the mean vector is obtained for each j-th task as the sample average of the $\bar{W}$ sample sets $D_{j-\bar{W}}, D_{j-\bar{W}+1}, \ldots, D_{j-1}$, then the MSE is given by

$$\boldsymbol{s}_{j,\bar{W}}^{j-1} = \sum_{i=j-\bar{W}+1}^{j} \frac{(i-j+\bar{W})^2}{\bar{W}^2} \boldsymbol{d}_i + \sum_{i=j-\bar{W}}^{j-1} \frac{\boldsymbol{s}_i}{\bar{W}^2}$$

and the ESS satisfies

$$n_{j,\bar{W}}^{j-1} \geq n \frac{6\bar{W}}{(\bar{W}+1)(2\bar{W}+1)nd + 6} \tag{1.16}$$

with $1 \leq \bar{W} \leq j-1$ and $\boldsymbol{\sigma}_j, d$, and n as in Theorem 3.

Analogously to the ESS of the proposed methodology in (1.14), the ESS of each task using sliding windows increases with the sample size n and decreases with the expected quadratic change d. Figure 1.3 illustrates the increase in ESS with respect to the sample size due to the proposed methodology in comparison with techniques sliding windows techniques. Specifically, such figure shows the ESS of the proposed methods with $k = 50$ tasks and the ESS using sliding windows in (1.16) with $\bar{W} = 5, 25$, and 45. The ESS significantly increases when nd decreases between 1 and $1/k^2$ for the proposed methodology and between 1 and $1/\bar{W}^2$ using sliding windows.

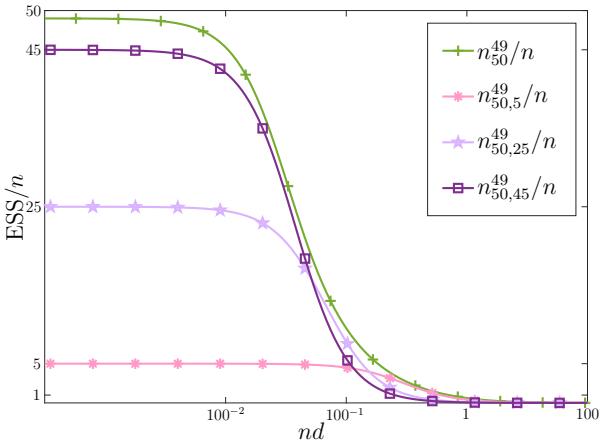


Figure 1.3: ESS of the proposed methodology in comparison with ESS of sliding windows.

The following result provides bounds for the minimax risk for each j-th task with respect to the smallest risk leveraging information from all the tasks in the sequence.

Theorem 4. Let M, n_j^j, and κ be as in Theorem 2 for any $j \in \{1, 2, ..., k\}$. Then, with probability at least $1 - \delta$, we have that

$$R(\mathcal{U}_j^k) \leq R_j^\infty + \frac{M(\kappa+1)\sqrt{2\log(2m/\delta)}}{\sqrt{n_j^k}} \left\|\boldsymbol{\mu}_j^\infty\right\|_1 \tag{1.17}$$

with

$$n_j^k \geq \frac{\left(\|\boldsymbol{\sigma}_j^2\|_\infty + n_j^j \|\boldsymbol{d}_{j+1}\|_\infty\right)^2 n_{j+1}^k}{\|\boldsymbol{\sigma}_j^2\|_\infty^2 + \|\boldsymbol{d}_{j+1}\|_\infty (\|\boldsymbol{\sigma}_j^2\|_\infty + n_j^j \|\boldsymbol{d}_{j+1}\|_\infty) n_{j+1}^k}$$

for $j < k$.

Proof. See Appendix A.6. □

Theorem 4 provides performance guarantees leveraging information from all the tasks in the sequences. The ESS of each task n_j^k is given by the ESS n_j^j of the j-th task obtained leveraging information from the j previous tasks and the ESS n_{j+1}^k of the $(j+1)$-th task leveraging information from the k tasks. Such ESS n_j^k is significantly greater than that obtained using information only from preceding tasks n_j^j because $n_{j+1}^k \geq n_j^j$. In particular, if $\boldsymbol{d}_{j+1}$ is large, the ESS is given by n_j^j, while if $\boldsymbol{d}_{j+1}$ is small, the ESS is given by n_{j+1}^k.

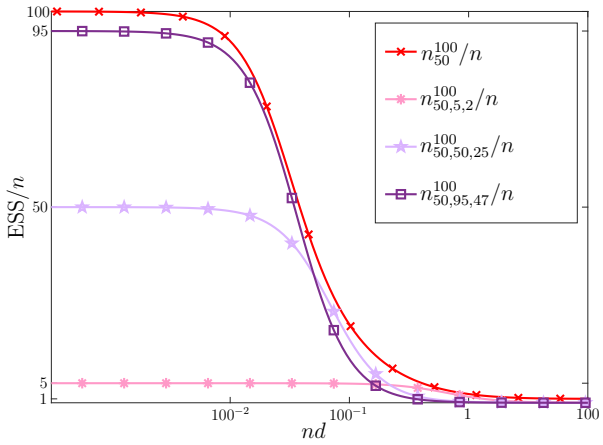


Figure 1.4: ESS increase provided by the proposed methodology in comparison with ESS of sliding windows.

Theorem 4 shows the increase in ESS in terms of the ESS of consecutive tasks. The following result allows to directly quantify the ESS in terms of the sample size and the expected quadratic change.

Theorem 5. Let d, $\boldsymbol{\sigma}_j$, and n be such that $d \geq \|\boldsymbol{d}_j\|_\infty$, $\|\boldsymbol{\sigma}_j^2\|_\infty \leq 1$, and $n \leq n_j$ for $j = 1, 2, \ldots, k$. For any $j \in \{1, 2, \ldots, k\}$, we have that the ESS in (1.17) satisfies

$$n_j^k \geq n\left(1 + \frac{(1+\alpha)^{2j-1} - 1 - \alpha}{\alpha(1+\alpha)^{2j-1} + \alpha} + \frac{(1+\alpha)^{2(k-j)+1} - 1 - \alpha}{\alpha(1+\alpha)^{2(k-j)+1} + \alpha}\right) \tag{1.18}$$

with $\alpha = \frac{2}{\sqrt{1+\frac{4}{nd}}-1}$. In particular, if $nd < \frac{1}{j^2}$, we have that $n_j^k \geq n_j^j + n\frac{j(k-j)}{j+2(k-j)} \geq n\left(1 + \frac{j-1}{3} + \frac{j(k-j)}{j+2(k-j)}\right)$ for $j > 1$ with n_j^j given by (1.15).

Proof. See Appendix A.7. □

The above theorem characterizes the increase in ESS provided the proposed methodology in terms of the tasks' expected quadratic change. Such increase grows monotonically with the number of previous tasks j and with the number of tasks $k - j$ after j. In addition, it becomes proportional to the total number of tasks k when the expected quadratic change is smaller than $1/(j^2 n)$ and $j \geq k/2$.

Techniques based on sliding windows can adapt to changes in the distribution leveraging information from all the tasks in the sequence. If the mean and confidence vectors are obtained for each j-th task with $j \in \{1, 2, \ldots, k\}$ as the sample average of the $\bar{W}$ closest sample sets $D_{j-\bar{W}+\widehat{W}+1}, D_{j-\bar{W}+\widehat{W}+2}, \ldots, D_j, \ldots, D_{j+W_l-\widehat{W}}$ then, the

MSE is given by

$$\boldsymbol{s}^k_{j,\bar{W},\widehat{W}} = \sum_{i=j-\widehat{W}+2}^{j} \frac{(i-j+\widehat{W}-1)^2}{\bar{W}^2}\boldsymbol{d}_i + \sum_{i=j+1}^{j+\bar{W}-\widehat{W}} \frac{(j+\bar{W}-\widehat{W}+1-i)^2}{\bar{W}^2}\boldsymbol{d}_i + \sum_{i=j-\widehat{W}+1}^{j+\bar{W}-\widehat{W}} \frac{\boldsymbol{s}_i}{\bar{W}^2}$$

and the ESS satisfies $n^k_{j,\bar{W},\widehat{W}} \geq n\frac{6\bar{W}}{(6\widehat{W}^2+2\bar{W}^2+3\bar{W}+1-6\widehat{W}\bar{W}-6\widehat{W})nd+6}$ with $1 \leq \widehat{W} \leq \bar{W} \leq k$ and $\boldsymbol{\sigma}_j, d$, and n as in Theorem 3.

Figure 1.4 illustrates the increase in ESS with respect to the sample size due to the proposed methods in comparison with sliding windows. Specifically, such figure shows the ESS of the proposed methods and the ESS of the sliding windows in (1.16) using centered windows with $\bar{W} = 5, 50$, and 95.

1.6 Implementation in scenarios under time-dependent environments

This section describes the implementation and the computational and memory complexity of the proposed methodology in batch learning scenarios including MDA and MTL and online learning scenarios including SCD and CL.

1.6.1 Batch learning scenarios

In MDA, learning methods use k sample sets corresponding with different tasks (source domains) to obtain a classification rule for the k-th task (target domain) with $|D_k| > 0$ [21–23] or $|D_k| = 0$ [1,9,88–90]. Algorithm 1 details the implementation of the proposed methodology for MDA that obtains mean and MSE vectors $\boldsymbol{\tau}^k_k$ and $\boldsymbol{s}^k_k$ as in (1.2)-(1.3) if $|D_k| > 0$ and mean and MSE vectors $\boldsymbol{\tau}^{k-1}_k$ and $\boldsymbol{s}^{k-1}_k$ as in (1.5)-(1.6) if $|D_k| = 0$. Then, we take the confidence vector as in (7) and obtain the classifier parameter and the minimax risk for the k-th task solving (6) (see Alg. 8 in Section 2.4). Algorithm 1 has computational complexity $O(km + n(2^{|\mathcal{Y}|} - 1)Km)$ and memory complexity $O(m)$ where m is the length of the feature mapping, n is the sample size, and K is the number of iterations of the optimization step.

In MTL, learning methods use k sample sets corresponding with different tasks to obtain classification rules $\{\mathrm{h}^k_j\}_{1\leq j\leq k}$ for the k tasks [3, 4, 30, 31]. Algorithm 2 details the implementation of the proposed methodology for MTL that first obtains mean and MSE vectors $\{\boldsymbol{\tau}^k_j\}_{1\leq j\leq k}$ and $\{\boldsymbol{s}^k_j\}_{1\leq j\leq k}$ as in (1.8)-(1.9). Then, we take the confidence vectors $\{\boldsymbol{\lambda}^k_j\}_{1\leq j\leq k}$ as in (7) and obtain the classifier parameters $\{\boldsymbol{\mu}^k_j\}_{1\leq j\leq k}$ and the minimax risks $\{R(\mathcal{U}^k_j)\}_{1\leq j\leq k}$ for the k tasks in the sequence solving (6) (see Alg. 8).

Algorithm 1 MDA

Input: $D_1, D_2, \ldots, D_k$
Output: $\boldsymbol{\mu}_k^k, R(\mathcal{U}_k^k)$ if $|D_k| > 0$ and $\boldsymbol{\mu}_k^{k-1}, R(\mathcal{U}_k^{k-1})$ if $|D_k| = 0$.
if $|D_k| > 0$ **then**
  for $j = 1, 2, ..., k-1$ **do**
    Obtain mean and MSE vectors $\boldsymbol{\tau}_j^j, \boldsymbol{s}_j^j$ as in (1.2)-(1.3)
  Take confidence vector $\boldsymbol{\lambda}_k^k$ as $\sqrt{\boldsymbol{s}_k^k}$
  Obtain classifier parameter and minimax risk $\boldsymbol{\mu}_k^k$ and $R(\mathcal{U}_k^k)$ solving (6)
else
  for $j = 1, 2, ..., k-1$ **do**
    Obtain mean and MSE vectors $\boldsymbol{\tau}_j^{j-1}, \boldsymbol{s}_j^{j-1}$ as in (1.5)-(1.6)
  Take confidence vector $\boldsymbol{\lambda}_k^{k-1}$ as $\sqrt{\boldsymbol{s}_k^{k-1}}$
  Obtain classifier parameter and minimax risk $\boldsymbol{\mu}_k^{k-1}$ and $R(\mathcal{U}_k^{k-1})$ solving (6)

Algorithm 2 has computational complexity $\mathcal{O}(mk + n(2^{|\mathcal{Y}|} - 1)Kmk)$ and memory complexity $\mathcal{O}(mk + k)$.

Algorithm 2 MTL

Input: $D_1, D_2, \ldots, D_k$
Output: $\{\boldsymbol{\mu}_j^k\}_{1 \leq j \leq k}$, and $\{R(\mathcal{U}_j^k)\}_{1 \leq j \leq k}$
for $j = 1, 2, ..., k$ **do**
  Obtain mean and MSE vectors $\boldsymbol{\tau}_j^j, \boldsymbol{s}_j^j$ as in (1.2)-(1.3)
for $j = k-1, k-2, ..., 1$ **do**
  Obtain mean and MSE vectors $\boldsymbol{\tau}_j^k, \boldsymbol{s}_j^k$ as in (1.8)-(1.9)
  Take confidence vector $\boldsymbol{\lambda}_j^k$ as $\sqrt{\boldsymbol{s}_j^k}$
  Obtain classifier parameter and minimax risk $\boldsymbol{\mu}_j^k$ and $R(\mathcal{U}_j^k)$ solving (6)

1.6.2 Online learning scenarios

In SCD, sample sets corresponding with different tasks arrive over time and learning methods use for each k-th task the most recent sample set D_{k-1} to obtain the classification rule h_k^{k-1} [43–45]. Algorithm 3 details the implementation of the proposed methodology for SCD that first obtains mean and MSE vectors $\boldsymbol{\tau}_k^{k-1}$ and $\boldsymbol{s}_k^{k-1}$ that are updated as in (1.5)-(1.6) from those for the previous task $\boldsymbol{\tau}_{k-1}^{k-2}$ and $\boldsymbol{s}_{k-1}^{k-2}$ together with sample set D_{k-1}. Then, we take the confidence vector $\boldsymbol{\lambda}_k^{k-1}$ as in (7) and obtain the classifier parameter $\boldsymbol{\mu}_k^{k-1}$ and the minimax risk $R(\mathcal{U}_k^{k-1})$ for the k-th task solving (6) (see Alg. 8). Algorithm 3 has at each step k computational complexity $O(m+n(2^{|\mathcal{Y}|}-1)Km)$ and memory complexity $O(m)$.

Algorithm 3 SCD at step k

Input: D_{k-1}, $\boldsymbol{\tau}_{k-1}^{k-2}, \boldsymbol{s}_{k-1}^{k-2}$, and $\boldsymbol{\mu}_{k-1}^{k-2}$
Output: $\boldsymbol{\tau}_{k}^{k-1}, \boldsymbol{s}_{k}^{k-1}, \boldsymbol{\mu}_{k}^{k-1}$, and $R(\mathcal{U}_{k}^{k-1})$
Obtain mean and MSE vectors $\boldsymbol{\tau}_{k}^{k-1}, \boldsymbol{s}_{k}^{k-1}$ as in (1.5)-(1.6)
Take confidence vector $\boldsymbol{\lambda}_{k}^{k-1}$ as $\sqrt{\boldsymbol{s}_{k}^{k-1}}$
Obtain classifier parameter and minimax risk $\boldsymbol{\mu}_{k}^{k-1}$ and $R(\mathcal{U}_{k}^{k-1})$ solving (6)

In CL, sample sets corresponding with different tasks arrive over time and learning methods use at each step k the sample set D_k to obtain classification rules $\{\mathrm{h}_j^k\}_{1\leq j\leq k}$ for the sequence of tasks [11, 15, 16]. Algorithm 4 details the implementation of the proposed methodology for CL that obtains at each step k mean and MSE vectors $\{\boldsymbol{\tau}_j^k\}_{k-b\leq j\leq k}$ and $\{\boldsymbol{s}_j^k\}_{k-b\leq j\leq k}$ as in (1.8)-(1.9) for b backward steps. Then, we take confidence vectors $\{\boldsymbol{\lambda}_j^k\}_{k-b\leq j\leq k}$ as in (7) and obtain classification rules $\{\mathrm{h}_j^k\}_{k-b\leq j\leq k}$ and minimax risks $\{R(\mathcal{U}_j^k)\}_{k-b\leq j\leq k}$ solving (6) (see Alg. 8). Algorithm 4 has at each step k computational complexity $\mathcal{O}((b+1)mk + bn(2^{|\mathcal{Y}|}-1)Km)$ and memory complexity $\mathcal{O}((b+k)m + n(2^{|\mathcal{Y}|}-1)mb)$. The number of backward steps $b = k - j$ can be taken to be rather small since the benefits of learning from succeeding tasks are achieved using only $b = 3$ backward steps in most of the situations.

Algorithm 4 CL at step k

Input: $D_k, \boldsymbol{\tau}_{k-1}^{k-1}, \boldsymbol{s}_{k-1}^{k-1}, \boldsymbol{\tau}_j, \boldsymbol{s}_j, \boldsymbol{\mu}_j^j$ for $k-b \leq j < k$
Output: $\boldsymbol{\mu}_j^k$ for $k-b+1 \leq j \leq k$, $\boldsymbol{\tau}_k, \boldsymbol{s}_k, \boldsymbol{\tau}_j^j, \boldsymbol{s}_j^j$
Obtain mean and MSE vectors $\boldsymbol{\tau}_k^k, \boldsymbol{s}_k^k$ using (1.2)-(1.3)
Take confidence vector $\boldsymbol{\lambda}_k^k = \sqrt{\boldsymbol{s}_k^k}$
Obtain classifier parameter $\boldsymbol{\mu}_k^k$ and minimax risk $R(\mathcal{U}_k^k)$ solving (6)
for $j = k-1, k-2, \ldots, k-b$ **do**
    Obtain mean and MSE vectors $\boldsymbol{\tau}_j^k, \boldsymbol{s}_j^k$ using (1.8)-(1.10)
    Take confidence vector $\boldsymbol{\lambda}_j^k$ as $\sqrt{\boldsymbol{s}_j^k}$
    Obtain classifier parameter $\boldsymbol{\mu}_j^k$ and minimax risk $R(\mathcal{U}_j^k)$ solving (6)

In the following, we extend the CL scenario described above to situations in which a new sample set can correspond with a previously learned task. A sample set D corresponding with a t-th task for $t \in \{1, 2, \ldots, k\}$ can arrive at any time step. Since the t-th task is a previously learned task, we first update the sample average and MSE vector $\boldsymbol{\tau}_t$ and $\boldsymbol{s}_t$ given by (7) of the t-th task using the new sample set; secondly, we update mean and MSE vectors $\boldsymbol{\tau}_j^j$ and $\boldsymbol{s}_j^j$ as in (1.2)-(1.3) for each j-th task with $j \in \{t, t+1, \ldots, k\}$; then, we update mean and MSE vectors $\boldsymbol{\tau}_j^k$ and $\boldsymbol{s}_j^k$ as in (1.8)-(1.9) for each j-th task with $j \in \{1, 2, \ldots, k\}$. Such mean and MSE vectors are the mean and MSE vectors obtained as in Section 1.4 using all samples corresponding with the t-th task at time step t. Then, taking $\boldsymbol{\sigma}_j^2 = \mathbb{V}\mathrm{ar}_{\mathrm{p}_j}\{\Phi(x, y)\}$ and $\boldsymbol{d}_j = \mathbb{E}\{\boldsymbol{w}_j^2\}$, the updated

Table 1.1: Dataset characteristics.

Dataset	Type	Samples	$\lvert\mathcal{Y}\rvert$	Tasks	Reference
Rotating hyperplane	Synthetic	30,000	2	100	[91]
BAF	Tabular	1,000,000	2	3,333	[92]
Elec2	Tabular	45,312	2	151	[57, 93]
Airlines	Tabular	539,383	2	1,797	[58, 93]
USPS	Tabular	2,930	2	9	[94]
Spam	Tabular	6,213	2	20	[95]
Power supply	Tabular	29,928	2	99	[6]
Yearbook	Images	37,921	2	10	[19]
ImageNet Noise	Images	12,000	2	10	[96, 97]
DomainNet	Images	6,256	4	6	[98]
UTKFace	Images	23,500	2	94	[99]
Rotated MNIST	Images	70,000	2	60	`http://yann.lecun.com/exdb/mnist/`
CLEAR	Images	10,490	3	10	[100]

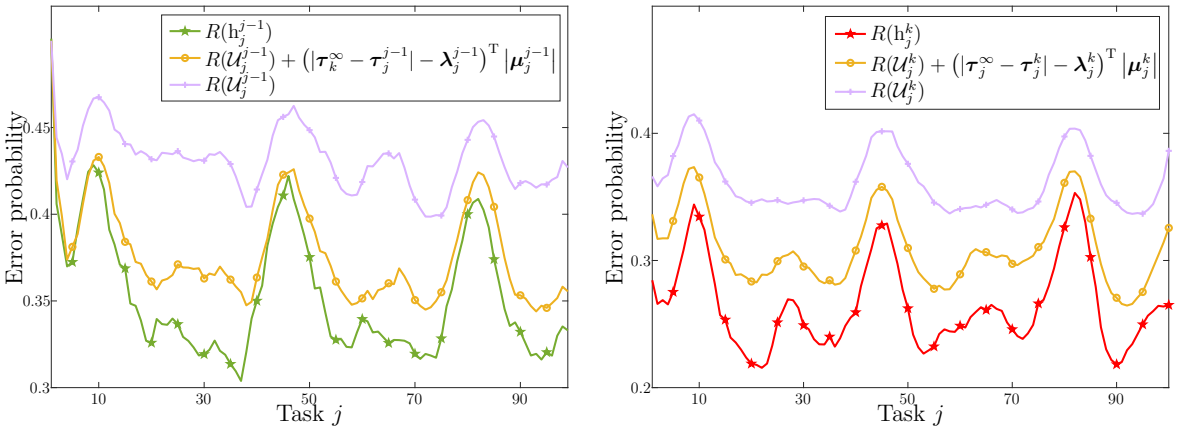
mean estimate $\boldsymbol{\tau}_j^k$ is the unbiased linear estimator that has the minimum MSE, and the updated $\boldsymbol{s}_j^k$ is its MSE.

1.7 Numerical results

This section evaluates the performance of the proposed methods in comparison with the presented performance guarantees and the state-of-the-art. In the first set of numerical results, we show the reliability of the performance guarantees; in the second set of numerical results, we show the improvement of the multidimensional adaptation; in the third set of numerical results, we show the performance in the batch learning scenarios MDA and MTL; and in the fourth set of numerical results, we show the performance in the online learning scenarios SCD and CL.

We utilize 13 public datasets with characteristics given in Table 1.1. The tabular datasets are divided in segments of 300 samples corresponding to consecutive times where each task corresponds to each of those segments; while the rest of the datasets are composed by time-dependent tasks (images with characteristics/quality/realism that change over time). The samples in each task are randomly splitted in 100 samples for test and the rest of samples for training and the samples used for training in the experiments are randomly sampled from each group of training samples in each repetition.

The results for the proposed methods are obtained using a feature mapping defined by multiple features over instances together with one-hot encodings of labels as (1). Such map is given by RFF with 200 Gaussian vectors and covariance matrix given by the scalar 10 for the tabular datasets [48, 74, 75], by the pixel values for the "Rotated MNIST" dataset, and by the last layer of the ResNet18 pre-trained network for the rest of image datasets [15, 101–103]. The confidence vector $\boldsymbol{\lambda}$ in equation (7) is estimated using $\lambda_0 = 0.7$ and vector $\boldsymbol{d}_j$ in equation (1.11) is estimated using $W = 2$. We use



(a) Bounds for error probabilities of each j-th task leveraging information from the $j-1$ previous tasks.

(b) Bounds for error probabilities of each j-th task leveraging information from the k tasks.

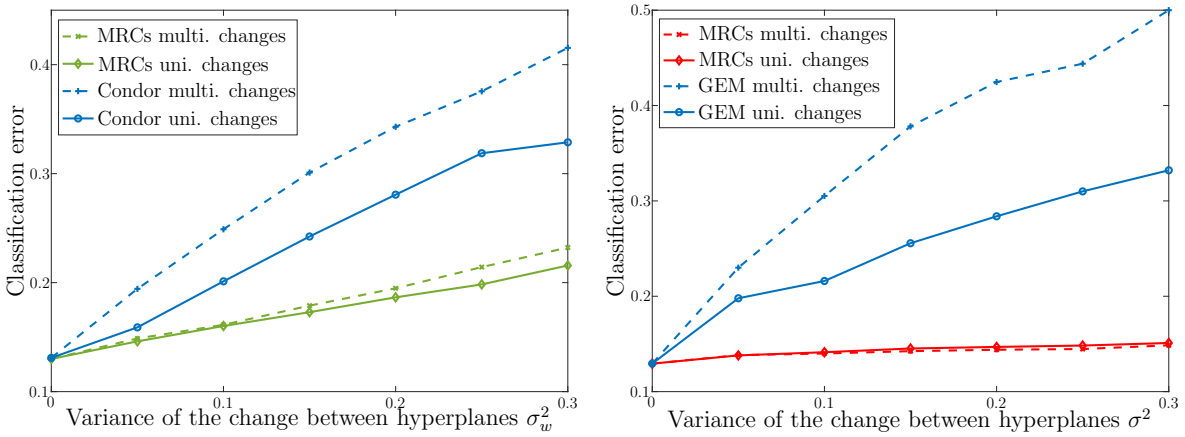
Figure 1.5: Results on synthetic data show the evolution over tasks of performance bounds and error probabilities.

the same hyper-parameters for all the results in this section for fair comparison with the state-of-the-art and to show that the methodology presented do not heavily rely on their values.

In the first set of numerical results we show the reliability of the presented bounds for error probabilities using the synthetic data since the error probability cannot be computed using real-world datasets. These numerical results are obtained averaging the classification errors and the bounds achieved with 10000 random instantiations of data samples in the synthetic data. Such data comprises a rotating hyperplane in 2 dimensional space where each coefficient of the hyperplane rotates 5 degrees between consecutive tasks.

Figures 1.5a and 1.5b show the averaged bounds for error probabilities corresponding to inequality (1.1) and the minimax risk in comparison with the true error probabilities. Figure 1.5a shows bounds for error probabilities of each j-th task $R(\mathrm{h}_j^{j-1})$ obtained leveraging information from sample sets $D_1, D_2, \ldots, D_{j-1}$ (MDA and SCD scenarios) and Figure 1.5b shows bounds for error probabilities of each j-th task $R(\mathrm{h}_j^{k})$ obtained leveraging information from sample sets $D_1, D_2, \ldots, D_k$ (MTL and CL scenarios). Such figures show that the bounds $R(\mathcal{U}_j^{j-1})$ and $R(\mathcal{U}_j^{k})$ can offer, for each j-th task, tight upper bounds for error probabilities $R(\mathrm{h}_j^{j-1})$ and $R(\mathrm{h}_j^{k})$, respectively.

In the second set of numerical results we show the improvement of multidimensional adaptation using the synthetic data since the tasks' changes are unknown in real-world datasets. These numerical results are obtained averaging the classification errors achieved with 100 random instantiations of data samples in the synthetic data with $n = 100$ samples per task. Such data comprises a rotating hyperplane in 5 dimensional space where each coefficient of the hyperplane changes between con-



(a) Classification error leveraging information from the $j-1$ previous tasks.

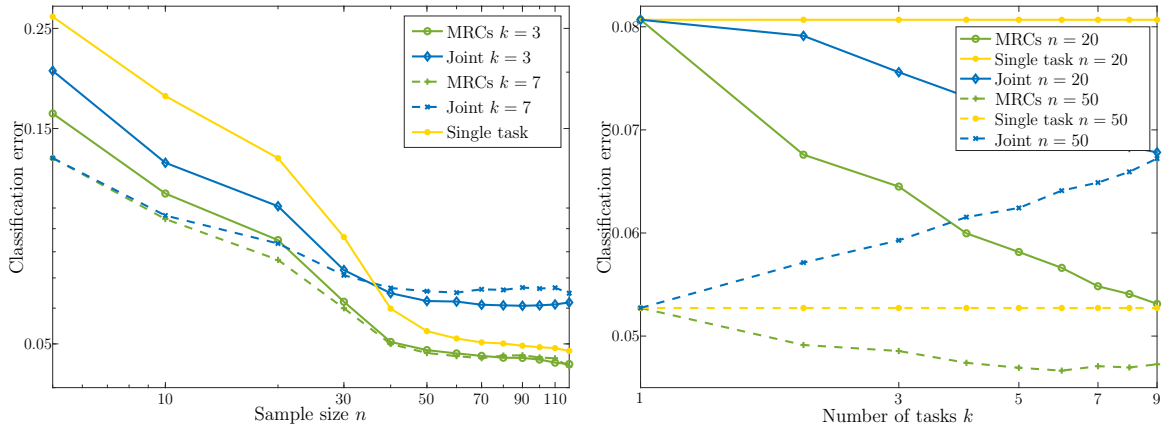
(b) Classification error leveraging information from the k tasks.

Figure 1.6: Results on synthetic data show the multidimensional adaptation to tasks' changes of the proposed methodology.

secutive tasks. Specifically, for each j-th task, we obtain the coefficients of the hyperplane $\mathbf{w}_j$ by adding a Gaussian random variable to the coefficients of the previous task as $\mathbf{w}_j = \mathbf{w}_{j-1} + N(\mathbf{0}, \sigma_w^2 \mathbf{I})$ for multidimensional (multi.) changes and as $\mathbf{w}_j = \mathbf{w}_{j-1} + \mathbf{1}N(0, \sigma_w^2)$ for unidimensional (uni.) changes where σ_w^2 denotes the variance of the change between consecutive hyperplanes. The results of the proposed methodology are compared with state-of-the-art techniques: Condor [104] that leverages information from previous tasks and gradient episodic memory (GEM) [11] that leverages information from all the tasks in the sequence.

Figures 1.6a and 1.6b show the classification error of the proposed methodology and state-of-the-art techniques increasing the variance of the change between consecutive tasks. Such figures show the performance improvement due to the multidimensional adaptation in comparison with state-of-the-art techniques leveraging information from previous tasks (MDA and SCD scenarios) and from all the tasks in the sequence (MTL and CL scenarios). Figure 1.6 shows that the proposed methodology better account for multidimensional tasks' changes than state-of-the-art techniques.

In the third set of numerical results we show the relationship among classification error, number of tasks, and sample size in batch learning scenarios using real-world datasets. These numerical results are obtained averaging, for each number of tasks and sample size, the classification errors achieved with 10 random instantiations of data samples in "Yearbook" dataset. The performance improvement of the proposed methodology is compared with relevant baselines for learning from a sequence of tasks: joint learning (also known as offline learning) [97, 105] and single-task learning (also known as independent learning) [49]. Joint learning and single-task learning obtain classification rules as in standard supervised classification using the samples from all the tasks and using samples only from the corresponding task, respectively.



(a) Classification error for MDA per sample size.

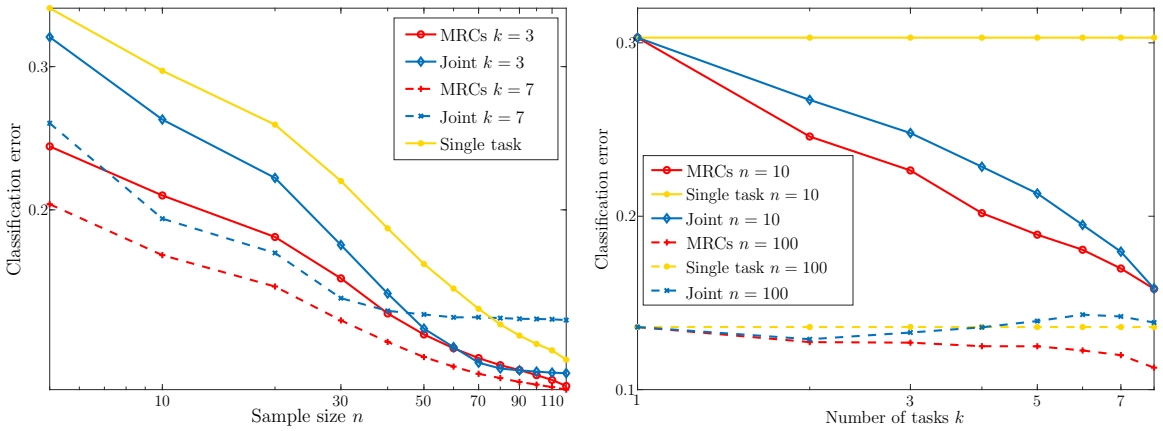
(b) Classification error for MDA per number of tasks.

Figure 1.7: Results on "Yearbook" dataset show the relationship among classification error, number of tasks, and sample size for MDA.

Figures 1.7a and 1.7b show the classification error for MDA for different sample sizes and number of tasks. Such figures show that the proposed learning methodology achieves significantly better results than joint learning and single-task learning. In particular, Figure 1.7a shows that the proposed methodology for $k = 3$ using $n = 20$ samples per task achieves similar results than joint learning for $k = 7$ using $n = 20$ samples per tasks and than single task learning using $n = 30$ samples per task. In addition, Figure 1.7b shows that the proposed methodology increases performance increasing the number of tasks using $n = 20$ and $n = 50$ samples per task, while the performance of single task learning remains constant increasing the number of tasks and the performance of joint learning decreases using $n = 50$ samples per task.

Figures 1.8a and 1.8b show the classification error for MTL for different sample sizes and number of tasks. Such figures show that the proposed learning methodology achieves significantly better results than joint learning and single-task learning. In particular, Figure 1.8a shows that the proposed methodology for $k =$ using $n = 40$ samples per task achieves similar results that joint learning for $k = 7$ using $n = 40$ samples per task and than single task learning using $n = 70$ samples per task. In addition, Figure 1.8b shows that the proposed methodology can improve performance as tasks arrive. The methods proposed can effectively adapt to tasks' changes that results in enhanced classification error.

In the fourth set of numerical results, we compare the performance of the proposed methdology for online learning scenarios with the state-of-the-art techniques for $n = 10$ and $n = 100$ samples per task. These numerical results are obtained computing the average classification error over all the tasks in 100 random instantiations of data samples. The proposed methodology is compared with 3 state-of-the-art SCD techniques: Condor [104], DriftSurf [6], and accuracy updated ensemble (AUE) [45]; and 4



(a) Classification error for MTL per sample size.

(b) Classification error for MTL per number of tasks.

Figure 1.8: Results on "Yearbook" dataset show the relationship among classification error, number of tasks, and sample size for MTL.

Table 1.2: Classification error of the proposed methodology in comparison with the state-of-the-art techniques SCD.

Dataset	BAF		Elec2		Airlines		USPS		Spam		Power supply	
n	10	100	10	100	10	100	10	100	10	100	10	100
Condor	1.11	1.12	38.90	40.10	43.32	43.42	48.52	48.01	**24.72**	26.25	**34.3**	33.00
Drift Surf	1.08	0.96	42.90	43.66	44.41	45.66	**38.00**	38.00	33.68	32.17	46.0	43.30
AUE	1.06	1.05	42.38	43.34	44.54	45.74	43.60	38.80	27.38	29.70	46.0	43.00
MRCs	**0.59**	**0.59**	**38.83**	**38.29**	**39.07**	**38.74**	40.61	**34.66**	26.23	**20.80**	40.26	**28.99**

state-of-the-art CL techniques: GEM [11], meta-experience replay (MER) [49], efficient continual learning algorithm (ELLA) [7], and elastic weight consolidation (EWC) [14]. The hyper-parameters in these methods are set to the default values provided by the authors.

Tables 1.2 and 1.3 show the classification error of the state-of-the-art techniques using the 12 real-world datasets described above. Such tables show that the proposed methodology for SCD and CL offers an overall improved performance compared to existing techniques. The proposed methodology can significantly improve performance in time-dependent environments with respect to the state-of-the-art.

Figure 1.9 shows the performance improvement receiving samples of the $j = 3$-th task at step $k = 7$. These numerical results are obtained averaging the classification errors for CL scenarios achieved with 100 random instantiations of data samples in "Yearbook" dataset. Figure 1.9 shows that receiving samples from a previously learned task not only improves performance of the corresponding task but also of all the tasks in the sequence. Figure 1.9 also shows the positive backward transfer of the proposed methodology for CL.

Table 1.3: Classification error of the proposed methodology in comparison with the state-of-the-art techniques for CL.

Dataset	Yearbook		ImageNet noise		DomainNet		UTKFaces		Rotated MNIST		CLEAR	
n	10	100	10	100	10	100	10	100	10	100	10	100
GEM	43.53	23.45	39.09	13.78	69.78	53.60	12.20	12.10	45.28	32.02	56.60	8.60
MER	38.62	19.37	27.25	12.71	47.58	30.26	12.13	12.13	**36.34**	34.54	20.53	7.40
ELLA	46.36	42.98	48.75	47.11	67.15	67.35	19.10	17.79	48.13	47.96	61.15	60.43
EWC	48.94	37.17	45.75	30.68	50.03	41.17	12.13	12.13	47.05	39.93	62.73	38.53
MRCs	**14.18**	**9.42**	**14.48**	**8.68**	**33.81**	**24.53**	**10.32**	**10.32**	37.02	**21.04**	**8.22**	**4.06**

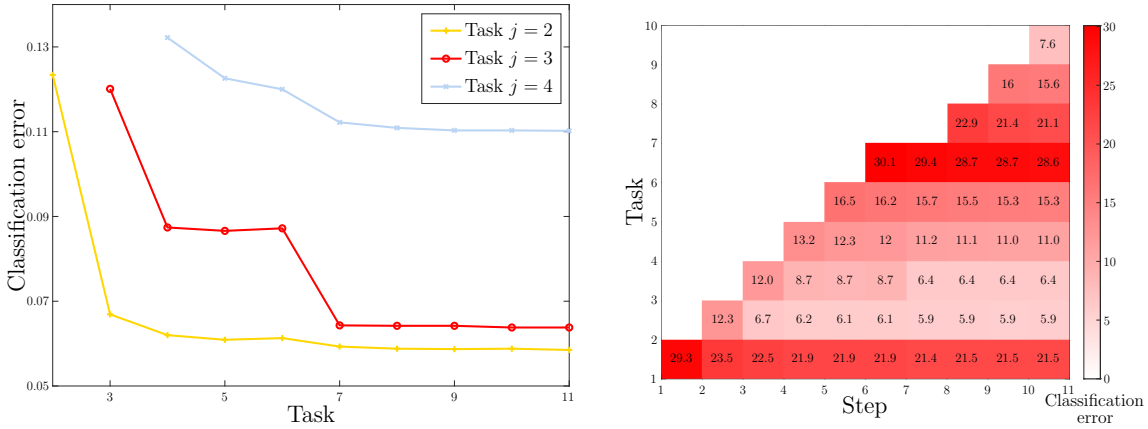


Figure 1.9: Results on "Yearbook" dataset show the classification error for CL receiving samples of the j-th task at step k with $j = 3$, $k = 7$.

1.8 Conclusion

This chapter proposes a learning methodology for time-dependent environments that includes multiple machine learning scenarios, accounts for multidimensional tasks' changes, and provides computable performance guarantees. The proposed methodology accounts for multidimensional adaptation to changes between consecutive tasks by estimating multiple statistical characteristics of the underlying distribution. In addition, we analytically characterize the increase in ESS achieved by the proposed methodology in terms of the expected quadratic change and the number of tasks. The numerical results assess the reliability of the performance guarantees presented and show the performance improvement in multiple machine learning scenarios using multiple datasets, sample sizes, and number of tasks. The proposed methodology can improve performance in a wide range of scenarios using efficient algorithms for learning in time-dependent environments.

Chapter 2

Minimax Classification under concept drift

2.1 Introduction

The statistical characteristics describing the underlying distribution of instance-label pairs often change with time in practical scenarios of supervised classification [17, 106]. Such concept drift is common in multiple applications including electricity price prediction [107], spam mail filtering [18], and credit card fraud detection [20]. For instance, in the problem of predicting electricity price increases/decreases, the statistical characteristics related to electricity demand, generation, and price often change over time due to varying habits and weather. Supervised classification in those scenarios is commonly referred to as learning under concept drift (e.g., [91, 107]), learning in a drifting (dynamic) scenario (e.g., [48, 72]), and online adaptive learning (e.g., [5, 108]).

Supervised classification techniques adapt to concept drift by updating classification rules as new instance-label pairs arrive. Conventional learning techniques account for a scalar rate of change by means of a carefully chosen parameter such as a learning rate [46, 47], forgetting factor [62, 63], or window size [57, 58]. Specifically, a slow/fast rate of change is tackled by using a low/high learning rate, forgetting factor, or window size. More sophisticated techniques account for a time-varying scalar rate of change by adjusting the learning rates [48], the forgetting factors [63], or the window sizes [57] over time. In particular, techniques based on dynamic regret minimization utilize a time-varying combination of rules obtained with different learning rates [5, 48]. However, in common scenarios, the concept drift cannot be adequately addressed accounting only for a scalar rate of change. Such inadequacy is due to the fact that time changes are commonly multidimensional, i.e., different statistical characteristics of instance-label pairs often change in a different manner (see Fig. 2.1). For instance, in the problem of predicting electricity price increases/decreases, the statistical characteristics related to demand often change differently from those related to generation.

Conventional techniques based on statistical learning and empirical risk minimiza-

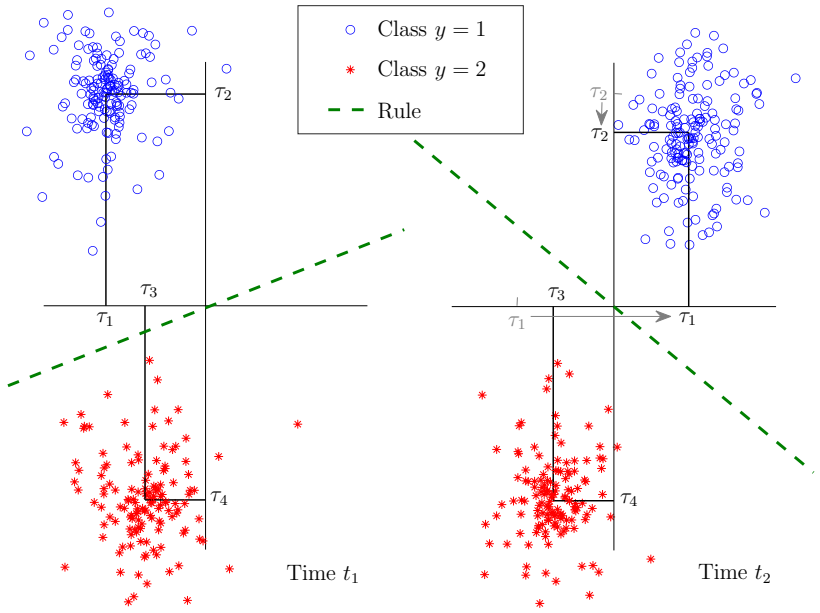


Figure 2.1. From time t_1 to time t_2, the statistical characteristic corresponding with class 1 (τ_1 and τ_2) significantly change, while those corresponding with class 2 (τ_3 and τ_4) do not change. AMRCs account for such multidimensional time changes by tightly tracking the time-varying underlying distribution.

tion do not provide performance guarantees under concept drift since instance-label pairs follow a time-varying underlying distribution. Techniques based on online learning and regret minimization provide performance guarantees for accumulated mistakes in terms of dynamic regret bounds [5, 47, 61, 108]. Furthermore, techniques based on statistical learning for the drifting scenario provide performance guarantees for the instantaneous error probability at specific times in terms of discrepancies between consecutive distributions [71,72]. However, the existing techniques offer qualitative bounds but not computable tight performance guarantees.

This chapter presents adaptive minimax risk classifiers (AMRCs) that account for multidimensional time changes and provide tight performance guarantees. Specifically, the main contributions presented in the chapter are as follows:

- We develop the learning methodology of AMRCs that provides multidimensional adaptation by estimating multiple statistical characteristics of the time-varying underlying distribution.
- We show that AMRCs can provide computable tight performance guarantees under concept drift in terms of instantaneous error probabilities and accumulated mistakes.
- We propose techniques to track the time-varying underlying distribution using dynamical systems that model multivariate and high-order changes in statistical characteristics.

- We propose techniques to learn AMRCs using an efficient subgradient method that utilizes warm-starts and maintains local approximations of the objective function.
- Using multiple benchmark datasets, we quantify the classification improvement of AMRCs in comparison with the state-of-the-art and numerically assess the reliability of the performance guarantees presented.

The rest of the chapter is organized as follows. Section 2.2 briefly describes the problem formulation and presents the learning methodology of AMRCs together with their performance guarantees. We propose sequential techniques for tracking and efficienct learning in Sections 2.3 and 2.4. Section 2.5 assesses the proposed methods using synthetic and benchmark datasets.

Notations: calligraphic letters represent sets; bold lowercase letters represent vectors; bold capital letters represent matrices; $\mathbf{I}$ denotes the identity matrix; $\mathbb{1}\{\cdot\}$ denotes the indicator function; $\mathrm{sgn}(\cdot)$ denotes the sign function; $\|\cdot\|_1$ and $\|\cdot\|_\infty$ denote the 1-norm and the infinity norm of its argument; $\otimes$ denotes the Kronecker product; $(\,.\,)_+$ denotes the positive part of its argument; $[\,\cdot\,]^{\mathrm{T}}$ denotes the transpose of its argument; $\preceq$ and $\succeq$ denote vector inequalities; $\mathbb{E}_{\mathrm{p}}\{\,\cdot\,\}$ denotes the expectation of its argument with respect to distribution p; and $\mathbf{e}_i$ denotes a vector with one on the i-th component and zeros in the remaining components.

2.2 Methodology of adaptive MRCs

This section briefly describes the problem formulation and presents the learning methodology of AMRCs. Then, we describe how AMRCs provide multidimensional adaptation and tight performance guarantees.

In supervised classification under concept drift, samples arrive over time and the instance-label pair (x_t, y_t) at time t is a sample from an underlying distribution $\mathrm{p}_t \in \Delta(\mathcal{X} \times \mathcal{Y})$ that changes over time. Learning methods use each new pair to update the previous classification rule. Specifically, at each time t, these techniques predict a label $\hat{y}_t$ corresponding with a new instance x_t using the rule h_t at time t, then they obtain updated rule h_{t+1} when the true label y_t is provided.

Figure 2.2 depicts the methodology of AMRCs that sequentially update

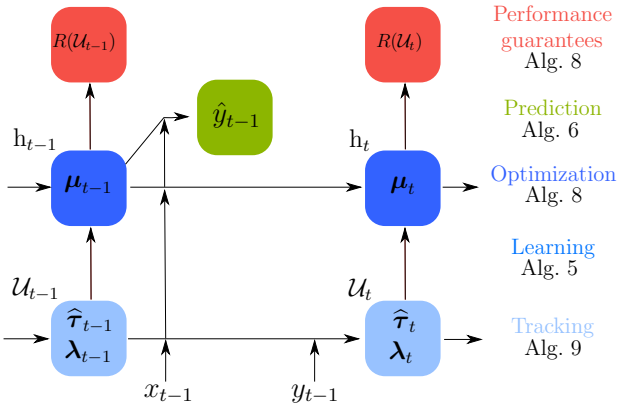


Figure 2.2. Diagram for the methodology of AMRCs.

minimax risk classifier (MRC) rules as
new instance-label pairs arrive, and Algorithms 5 and 6 specify the learning and prediction stages of AMRCs. At learning, AMRCs adapt to multidimensional time changes by tightly tracking the time-varying underlying distribution (see Alg. 7); optimize the classifier parameters by using an efficient subgradient method (see Alg. 8); and obtain performance guarantees by using the minimax risk (see Alg. 8 and Section 2.2.2).

The tracking step updates at each time the uncertainty set $\mathcal{U}_t$ using the uncertainty set at previous time $\mathcal{U}_{t-1}$. Specifically, the uncertainty set $\mathcal{U}_t$ is determined by the mean and confidence vectors $\boldsymbol{\tau}_t$ and $\boldsymbol{\lambda}_t$ that are updated from those at time $t-1$ together with the instance-label pair (x_{t-1}, y_{t-1}). The optimization step updates at each time the classification rule h_t and obtains the minimax risk $R(\mathcal{U}_t)$ using the updated uncertainty set $\mathcal{U}_t$ together with the rule at previous time h_{t-1}. Specifically, the classification rule h_t is determined by parameter $\boldsymbol{\mu}_t$ that is obtained from the updated mean and confidence vectors $\boldsymbol{\tau}_t$ and $\boldsymbol{\lambda}_t$ together with parameter $\boldsymbol{\mu}_{t-1}$.

Algorithm 5 Learning AMRCs

Input: (x_{t-1}, y_{t-1}), $\boldsymbol{\tau}_{t-1}, \boldsymbol{\lambda}_{t-1}$, and $\boldsymbol{\mu}_{t-1}$
Output: $\boldsymbol{\tau}_t, \boldsymbol{\lambda}_t$, $\boldsymbol{\mu}_t$, and $R(\mathcal{U}_t)$
Update $\boldsymbol{\tau}_t$ and $\boldsymbol{\lambda}_t$ using (x_{t-1}, y_{t-1}), $\boldsymbol{\tau}_{t-1}$, and $\boldsymbol{\lambda}_{t-1}$ (see Alg. 7)
Update $\boldsymbol{\mu}_t$ and obtain $R(\mathcal{U}_t)$ solving (6) using $\boldsymbol{\tau}_t, \boldsymbol{\lambda}_t$, and $\boldsymbol{\mu}_{t-1}$ (see Alg. 8)

Algorithm 6 Prediction with AMRCs

Input: x_t and $\boldsymbol{\mu}_t$
Output: $\hat{y}_t$ for AMRC h_t or for deterministic AMRC $\mathrm{h}_t^{\mathrm{d}}$
$c_x \leftarrow \sum_{y \in \mathcal{Y}} \left(\Phi(x_t, y)^{\mathrm{T}} \boldsymbol{\mu}_t - \varphi(\boldsymbol{\mu}_t)\right)_+$
if $c_x = 0$ **then**
  for $y \in \mathcal{Y}$ **do**
    $\mathrm{h}_t(y|x_t) \leftarrow 1/|\mathcal{Y}|$
else
  for $y \in \mathcal{Y}$ **do**
    $\mathrm{h}_t(y|x_t) \leftarrow \left(\Phi(x_t, y)^{\mathrm{T}} \boldsymbol{\mu}_t - \varphi(\boldsymbol{\mu}_t)\right)_+ / c_x$
Draw $\hat{y}_t$ from distribution $\mathrm{h}_t(y|x_t)$ or obtain $\hat{y}_t$ from $\arg\max_{y \in \mathcal{Y}} \mathrm{h}_t(y|x_t)$

2.2.1 Multidimensional adaptation to time changes

The proposed learning methodology can provide multidimensional adaptation to time changes because AMRCs estimate multiple statistical characteristics of the time-varying underlying distribution.

The proposed AMRCs estimate the evolution over time of the vector formed by the expectations of the feature mapping. At each time t, such mean vector $\boldsymbol{\tau}_t^{\infty} =$

$\mathbb{E}_{\mathrm{p}_t}\{\Phi(x,y)\} \in \mathbb{R}^m$ represents the statistical characteristics of the underlying distribution p_t as measured by the feature map $\Phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^m$. AMRCs account for situations in which different characteristics change in a different manner by estimating the evolution of each component in the mean vector.

For a feature mapping Φ given by one-hot encodings of labels as in (1), each of the $m = |\mathcal{Y}|d$ components of the mean vector represents one conditional expectation. For $i = 1, 2, ..., m$, the i-th component $\Phi_i(x, y)$ has associated a label $j = 1, 2, ..., |\mathcal{Y}|$ and an instance feature Ψ_r for $r = 1, 2, ..., d$, since

$$\Phi_i(x, y) = \mathbb{1}\{y = j\}\Psi_r(x)$$

with $i = (d-1)j + r$. Then, the i-th component of the mean vector $\boldsymbol{\tau}_t^\infty$ corresponding to the j-th label and the r-th instance feature is given by

$$\tau_{t,i}^\infty = \mathbb{E}_{\mathrm{p}_t}\{\Phi_i(x, y)\} = \mathrm{p}_t(y = j)\mathbb{E}_{\mathrm{p}_t}\{\Psi_r(x)|y = j\}$$

where $\mathrm{p}_t(y = j)$ denotes the probability of label $y = j$. The i-th component of the mean vector describes the probability of the j-th label together with the expected value of the r-th instance feature when the label takes its j-th value. In Section 2.3 we propose techniques for estimating each of the mean vector components accounting for multivariate and high-order time changes.

2.2.2 Performance guarantees

The proposed learning methodology can provide performance guarantees under concept drift because AMRCs minimize the worst-case error probability over distributions in a time-varying uncertainty set.

The proposed AMRCs provide performance guarantees in terms of the minimax risks obtained at learning and also with respect to the smallest minimax risks corresponding with the optimal minimax rules. Such rules correspond with uncertainty sets given by the true expectation of the feature mapping Φ, that is

$$\mathcal{U}_t^\infty = \{\mathrm{p} \in \Delta(\mathcal{X} \times \mathcal{Y}) : \mathbb{E}_{\mathrm{p}}\{\Phi(x, y)\} = \boldsymbol{\tau}_t\}. \tag{2.1}$$

The minimum worst-case error probability over distributions in $\mathcal{U}_t^\infty$ is given by

$$R_t^\infty = \min_{\boldsymbol{\mu}} 1 - \boldsymbol{\tau}_t^{\mathrm{T}}\boldsymbol{\mu} + \varphi(\boldsymbol{\mu}) = 1 - \boldsymbol{\tau}_t^{\mathrm{T}}\boldsymbol{\mu}_t^\infty + \varphi(\boldsymbol{\mu}_t^\infty) \tag{2.2}$$

corresponding with the AMRC given by parameters $\boldsymbol{\mu}_t^\infty$. Such classification rule is referred to as optimal minimax rule because for any uncertainty set $\mathcal{U}_t$ given by (2) that contains the underlying distribution, we have that $\mathcal{U}_t^\infty \subseteq \mathcal{U}_t$ and hence $R_t^\infty \leq R(\mathcal{U}_t)$. This optimal minimax rule could only be obtained by an exact estimation of the mean vector that in turn would require an infinite amount of instance-label pairs at each time.

The following result shows performance guarantees for AMRCs both in terms of instantaneous bounds for error probabilities and bounds for accumulated mistakes.

Theorem 6. For $t = 1, 2, ..., T$, let $\mathcal{U}_t$ be the uncertainty set given by $\boldsymbol{\tau} = \boldsymbol{\tau}_t$ and $\boldsymbol{\lambda} = \boldsymbol{\lambda}_t$ in (2), and $\hat{y}_t$ be a label provided by an AMRC h_t for $\mathcal{U}_t$ given by parameter $\boldsymbol{\mu}_t$. Then, we have that

$$R(\mathrm{h}_t) \leq R(\mathcal{U}_t) + \alpha_t \leq R_t^\infty + \beta_t \tag{2.3}$$

and with probability at least $1 - \delta$

$$\begin{aligned} \sum_{t=1}^{T} \mathbb{1}\left\{\hat{y}_t \neq y_t\right\} &\leq \sum_{t=1}^{T} R(\mathcal{U}_t) + \sum_{t=1}^{T} \alpha_t + \sqrt{2T \log \frac{1}{\delta}} \\ &\leq \sum_{t=1}^{T} R_t^\infty + \sum_{t=1}^{T} \beta_t + \sqrt{2T \log \frac{1}{\delta}} \end{aligned} \tag{2.4}$$

where α_t and β_t can be taken as

$$\begin{aligned} \alpha_t &= \left\| |\boldsymbol{\tau}_t^\infty - \boldsymbol{\tau}_t| - \boldsymbol{\lambda}_t \right\|_\infty \left\| \boldsymbol{\mu}_t \right\|_1 \\ \beta_t &= \left(\left\| \boldsymbol{\tau}_t^\infty - \boldsymbol{\tau}_t \right\|_\infty + \left\| \boldsymbol{\lambda}_t \right\|_\infty \right) \left\| \boldsymbol{\mu}_t^\infty - \boldsymbol{\mu}_t \right\|_1 \end{aligned}$$

for any $\boldsymbol{\lambda}_t \succeq \mathbf{0}$, and as

$$\alpha_t = 0,\ \beta_t = 2 \left\| \boldsymbol{\lambda}_t \right\|_\infty \left\| \boldsymbol{\mu}_t^\infty \right\|_1$$

for $\boldsymbol{\lambda}_t \succeq |\boldsymbol{\tau}_t^\infty - \boldsymbol{\tau}_t|$.

Proof. See B.1. □

Inequalities in (2.3) and (2.4) bound the instantaneous error probability and the accumulated mistakes of AMRCs, respectively. Inequalities in (2.3) are obtained as a generalization of bounds in [76, 77] to the addressed setting, while inequalities in (2.4) are obtained using the Azuma's inequality for the martingale difference $\mathbb{1}\{\hat{y}_t \neq y_t\} - R(\mathrm{h}_t)$. Note that the above inequalities also ensure generalization for deterministic AMRCs because $R(\mathrm{h}_t^{\mathrm{d}}) \leq 2R(\mathrm{h}_t)$ since $1 - \mathrm{h}_t^{\mathrm{d}}(y|x) \leq 2(1 - \mathrm{h}_t(y|x))$ for any $x \in \mathcal{X}, y \in \mathcal{Y}$.

Existing methods based on statistical learning for the drifting scenario provide bounds for instantaneous error probabilities with respect to empirical risks (generalization bound) [71, 72], while existing methods based on online learning and regret minimization provide bounds for accumulated mistakes with respect to that of comparator sequences (dynamic regret bound) [5, 47, 61, 108]. Theorem 6 for AMRC methods provides bounds for instantaneous error probabilities and accumulated mistakes with respect to the smallest minimax risk R_t^∞ corresponding to the optimal minimax rule (second inequalities in (2.3) and (2.4)).

The proposed AMRCs not only provide qualitative bounds but also computable tight performance guarantees given by the minimax risk obtained at learning. However, existing methods provide bounds in terms of quantities that are not computable at learning such as discrepancies between consecutive distributions [71, 72] and comparators' path-length [5, 108]. Theorem 6 for AMRC methods provides computable

tight bounds for instantaneous error probabilities and accumulated mistakes (first inequalities in (2.3) and (2.4), respectively) that are given by the minimax risk obtained by solving optimization (6) at learning. Specifically, the minimax risk $R(\mathcal{U}_t)$ directly provides such bounds if the error in the mean vector estimate is not underestimated, i.e., $\boldsymbol{\lambda}_t \succeq |\boldsymbol{\tau}_t^\infty - \boldsymbol{\tau}_t|$. In other cases, $R(\mathcal{U}_t)$ still provides approximate bounds as long as the underestimation $|\boldsymbol{\tau}_t^\infty - \boldsymbol{\tau}_t| - \boldsymbol{\lambda}_t$ is not substantial. In Section 2.5 we show that the presented bounds can provide tight performance guarantees for AMRCs in practice.

2.3 Tracking the time-varying underlying distribution

This section describes the proposed algorithms for tightly tracking the time-varying underlying distribution. Such algorithms utilize methods that are commonly used for target tracking and describe target trajectories using kinematic models (see e.g., [109]).

In what follows, we present techniques that estimate each component of the mean vector $\boldsymbol{\tau}_t^\infty$ at time t from instance-label pairs obtained up to time $t-1$. As described in Section 2.2.1, for $i = 1, 2, ..., m$, we denote by $\tau_{t,i}$ the i-th component of the mean vector that corresponds with the j-th label and the r-th instance feature, that is,

$$\tau_{t,i}^\infty = \mathrm{p}_t(y = j)\mathbb{E}_{\mathrm{p}_t}\{\Psi_r(x)|y = j\}$$

with $i = (d-1)j + r$ for $j = 1, 2, ..., |\mathcal{Y}|$ and $r = 1, 2, ..., d$. Then, the estimation of each component of the mean vector is obtained from the estimation of the corresponding label probability and instance feature conditional expectation. In particular, the j-th label probability for $j = 1, 2, ..., |\mathcal{Y}|$ is estimated using the W latest labels as

$$\widehat{\mathrm{p}}_t(y = j) = \frac{1}{W}\sum_{i=t-W}^{t-1} \mathbb{1}\{y_i = j\}. \tag{2.5}$$

The above conditional expectation denoted by $\gamma_{t,i}^\infty = \mathbb{E}_{\mathrm{p}_t}\{\Psi_r(x)|y = j\}$ for $i = 1, 2, ..., m$ is recursively estimated as described in the following.

We assume that $\gamma_{t,i}^\infty$ is k times differentiable with respect to time and denote by $\boldsymbol{\eta}_{t,i}^\infty \in \mathbb{R}^{k+1}$ the vector composed by $\gamma_{t,i}^\infty$ and its successive derivatives up to order k. As is usually done for target tracking [109], we model the evolution of such state vector $\boldsymbol{\eta}_{t,i}^\infty$ using the partially-observed linear dynamical system

$$\begin{aligned} \boldsymbol{\eta}_{t,i}^\infty &= \mathbf{H}_t\boldsymbol{\eta}_{t-1,i}^\infty + \mathbf{w}_{t,i} \\ \Phi_i(x_t, y_t) &= \gamma_{t,i}^\infty + v_{t,i}, \ \text{ if } y_t = j \end{aligned} \tag{2.6}$$

with transition matrix $\mathbf{H}_t = \mathbf{I} + \sum_{s=1}^k \Delta_t^s \mathbf{U}_s/s!$ where $\mathbf{U}_s$ is the $(k+1)\times(k+1)$ matrix with ones on the s-th upper diagonal and zeros in the rest of components, Δ_t is the time increment at t, $j = 1, 2, ..., |\mathcal{Y}|$, and $i = (d-1)j + r$ for $r = 1, 2, ..., d$. The variables

$\mathbf{w}_{t,i}$ and $v_{t,i}$ represent uncorrelated noise processes with mean zero and variance $\mathbf{Q}_{t,i}$ and $r_{t,i}^2$, respectively. Such variances can be estimated online using methods such as those proposed in [110, 111]. Dynamical systems as that given by (2.6) are known in target tracking as kinematic state models and can be derived using the k-th order Taylor expansion of $\gamma_{t,i}$ (see B.2 for a detailed derivation).

The result below allows to recursively obtain state vector estimates $\boldsymbol{\eta}_{t,i}$ with minimum mean squared error (MSE) together with their MSE matrices $\boldsymbol{\Sigma}_{t,i}$, for $i = 1, 2, ..., m$. Then, the first components of such vector and matrix provide the estimate and confidence for the conditional expectations $\gamma_{t,i}^{\infty}$ for $i = 1, 2, ..., m$.

Theorem 7. If the evolution of the state vector $\boldsymbol{\eta}_{t,i}^{\infty}$ is given by the dynamical system in (2.6), then the linear estimator of $\boldsymbol{\eta}_{t,i}^{\infty}$ based on $(x_1, y_1), (x_2, y_2), ..., (x_{t-1}, y_{t-1})$ that has the minimum MSE is given by the recursion

$$\boldsymbol{\eta}_{t,i} = \mathbf{H}_t \boldsymbol{\eta}_{t-1,i} - (\boldsymbol{\gamma}_{t-1,i} - \Phi_i(x_{t-1}, y_{t-1}))\mathbf{k}_{t,i} \tag{2.7}$$

$$\boldsymbol{\Sigma}_{t,i} = \mathbf{H}_t \boldsymbol{\Sigma}_{t-1,i} \mathbf{H}_t^{\mathrm{T}} + \mathbf{Q}_{t,i} - \mathbf{k}_{t,i} \mathbf{e}_1^{\mathrm{T}} \boldsymbol{\Sigma}_{t-1,i} \mathbf{H}_t^{\mathrm{T}} \tag{2.8}$$

where

$$\mathbf{k}_{t,i} = \mathbb{1}\{y_{t-1} = j\} \frac{\mathbf{H}_t \boldsymbol{\Sigma}_{t-1,i} \mathbf{e}_1}{\mathbf{e}_1^{\mathrm{T}} \boldsymbol{\Sigma}_{t-1,i} \mathbf{e}_1 + r_{t-1,i}^2}. \tag{2.9}$$

In addition, $\boldsymbol{\Sigma}_{t,i}$ is the MSE matrix of such estimator $\boldsymbol{\eta}_{t,i}$.

Proof. The unbiased linear estimator with minimum MSE for a dynamical system such as (2.6) is given by the Kalman filter recursions (see e.g., [112]). Then, equations (2.7) and (2.8) are obtained after some algebra from the Kalman recursions for predicted state vector and predicted MSE. □

The above theorem enables to tightly track the time-varying underlying distribution. Specifically, Theorem 7 allows to recursively obtain mean vector estimates $\boldsymbol{\tau}_t$ as well as its confidence vectors $\boldsymbol{\lambda}_t$ every time a new instance-label pair is received. Such vectors are obtained from the estimated label probabilities in equation (2.5) together with the estimated state vector and its MSE in recursions (2.7) and (2.8) as follows

$$\tau_{t,i} = \widehat{\mathrm{p}}_t(y = j)\, \mathbf{e}_1^{\mathrm{T}} \boldsymbol{\eta}_{t,i} = \widehat{\mathrm{p}}_t(y = j)\, \gamma_{t,i} \tag{2.10}$$

$$\begin{aligned} \lambda_{t,i}^2 =& \widehat{\mathrm{p}}_t(y = j)(1 - \widehat{\mathrm{p}}_t(y = j)) \mathbf{e}_1^{\mathrm{T}} \boldsymbol{\Sigma}_{t,i} \mathbf{e}_1 \\ &+ \widehat{\mathrm{p}}_t(y = j)^2 \mathbf{e}_1^{\mathrm{T}} \boldsymbol{\Sigma}_{t,i} \mathbf{e}_1 + \gamma_{t,i}^2 \widehat{\mathrm{p}}_t(y = j)(1 - \widehat{\mathrm{p}}_t(y = j)) \\ =& \widehat{\mathrm{p}}_t(y = j) \Big(\gamma_{t,i}^2 (1 - \widehat{\mathrm{p}}_t(y = j)) + \mathbf{e}_1^{\mathrm{T}} \boldsymbol{\Sigma}_{t,i} \mathbf{e}_1 \Big) \end{aligned} \tag{2.11}$$

with $i = (d-1)j + r$ for $j = 1, 2, ..., |\mathcal{Y}|$, $r = 1, 2, ..., d$.

The proposed techniques account for multivariate and high-order changes in statistical characteristics since the dynamical systems used model the detailed evolution of each component in the mean vector. Specifically, for $t = 1, 2, ...$ and $i = 1, 2..., m$, the i-th component of the mean vector estimate is updated at time t by using the

corresponding dynamical model in (2.6), the estimate at previous time, and the most recent instance-label pair. Such update accounts for the specific evolution at time t of the i-th component of the mean vector through the recursion in equation (2.7) and gain vector $\mathbf{k}_{t,i}$ in equation (2.9). In particular, updates for mean vector components and times with a low gain slightly change the estimate at previous time, while those updates for components and times with a high gain increase the relevance of the most recent instance-label pair.

Algorithm 7 details the proposed procedure to track the time-varying underlying distribution. Such algorithm has computational complexity $O(mk^3)$ and memory complexity $O(mk^2)$ where $m = |\mathcal{Y}|d$ is the length of the feature mapping and k is the order of the dynamical system in (2.6). In Section 2.5, we use dynamical systems with orders $k = 0, 1, 2$ which are known in target tracking as zero-order derivative models, white noise acceleration models, and Wiener process acceleration models, respectively.

Algorithm 7 Tracking of the time-varying underlying distribution

Input: $(x_{t-1}, y_{t-1}), \boldsymbol{\eta}_{t-1,i}, \boldsymbol{\Sigma}_{t-1,i}, \widehat{\mathrm{p}}_t(y = j), \mathbf{Q}_{t,i}$, and $r_{t,i}$ for $j = 1, 2, ..., |\mathcal{Y}|$ and $i = 1, 2, ..., m$

Output: $\boldsymbol{\tau}_t, \boldsymbol{\lambda}_t$, $\boldsymbol{\eta}_{t,i}$, and $\boldsymbol{\Sigma}_{t,i}$ for $i = 1, 2, ..., m$

for $i = 1, 2, ..., m$ **do**

Obtain state vector $\boldsymbol{\eta}_{t,i}$ using (2.7) and MSE $\boldsymbol{\Sigma}_{t,i}$ using (2.8)

Obtain the i-th component of mean vector estimate $\tau_{t,i}$ using (2.10) and confidence vector $\lambda_{t,i}$ using (2.11)

2.4 Efficient learning of AMRCs

This section describes the proposed algorithms that obtain AMRCs' parameters and minimax risks at each time. First, we describe the accelerated subgradient method (ASM) that solves the convex optimization problem (6) using Nesterov extrapolation strategy [79, 80]. Then, we propose efficient algorithms that use a warm-start for the ASM iterations and maintain a local approximation of the polyhedral function $\varphi(\cdot)$ in (5).

In what follows, we present techniques that efficiently obtain the classifier parameter $\boldsymbol{\mu}_t$ and minimax risk $R(\mathcal{U}_t)$ at time t from the classifier parameter $\boldsymbol{\mu}_{t-1}$ and the updated mean vector estimate $\boldsymbol{\tau}_t$ and confidence vector $\boldsymbol{\lambda}_t$. The ASM algorithm applied to optimization (6) obtains classifier parameters using the iterations for $l = 1, 2, ..., K$

$$\begin{aligned}\bar{\boldsymbol{\mu}}_t^{(l+1)} &= \boldsymbol{\mu}_t^{(l)} + a_l \left(\boldsymbol{\tau}_t - \partial\varphi(\boldsymbol{\mu}_t^{(l)}) - \boldsymbol{\lambda}_t \odot \operatorname{sign}(\boldsymbol{\mu}_t^{(l)})\right) \\ \boldsymbol{\mu}_t^{(l+1)} &= \bar{\boldsymbol{\mu}}_t^{(l+1)} + \theta_{l+1}(\theta_l^{-1} - 1)\left(\boldsymbol{\mu}_t^{(l)} - \bar{\boldsymbol{\mu}}_t^{(l)}\right)\end{aligned} \tag{2.12}$$

where $\boldsymbol{\mu}_t^{(l)}$ is the l-th iterate for $\boldsymbol{\mu}_t$, $\theta_l = 2/(l+1)$ and $a_l = 1/(l+1)^{3/2}$ are step sizes, and $\partial\varphi(\boldsymbol{\mu}_t^{(l)})$ denotes a subgradient of $\varphi(\cdot)$ at $\boldsymbol{\mu}_t^{(l)}$.

The proposed algorithm reduces the number of ASM iterations by using a warm-start that initializes the parameters $\boldsymbol{\mu}_t$ in (2.12) with the solution obtained at previous time $\boldsymbol{\mu}_{t-1}$. In addition, the ASM iterations are efficiently computed by maintaining a local approximation of the polyhedral function $\varphi(\cdot)$ in (5). Such function is given by the pointwise maximum of linear functions indexed by pairs of instances and labels' subsets $x \in \mathcal{X}, \mathcal{C} \subseteq \mathcal{Y}$. So that, if $\mathcal{I}$ is the set of such pairs we have that $\varphi(\boldsymbol{\mu})$ in (5) becomes

$$\varphi(\boldsymbol{\mu}) = \max_{i \in \mathcal{I}} \left\{ \mathbf{f}_i^{\mathrm{T}} \boldsymbol{\mu} - h_i \right\} \tag{2.13}$$

with $\mathbf{f}_i = \sum_{y \in \mathcal{C}} \Phi(x, y)/|\mathcal{C}| \in \mathbb{R}^m$ and $h_i = 1/|\mathcal{C}|$ for index i that corresponds to pair $(x, \mathcal{C})$. We use local approximations of (2.13) given by indices corresponding with the N most recent subgradients of $\varphi(\cdot)$. Specifically, if $\mathcal{R} \subseteq \mathbb{R}^m$ we have that for any $\boldsymbol{\mu} \in \mathcal{R}$,

$$\varphi(\boldsymbol{\mu}) = \max_{i \in \mathcal{J}_{\mathcal{R}}} \left\{ \mathbf{f}_i^{\mathrm{T}} \boldsymbol{\mu} - h_i \right\}$$

with

$$\begin{aligned} \mathcal{J}_{\mathcal{R}} &= \{ i \in \mathcal{I} : \exists\, \boldsymbol{\mu} \in \mathcal{R} \text{ with } \varphi(\boldsymbol{\mu}) = \mathbf{f}_i^{\mathrm{T}} \boldsymbol{\mu} - h_i \} \\ &= \{ i \in \mathcal{I} : \exists\, \boldsymbol{\mu} \in \mathcal{R} \text{ with } \mathbf{f}_i \in \partial\varphi(\boldsymbol{\mu}) \}. \end{aligned}$$

Then, if $\mathcal{J}_{\mathcal{R}} = \{i_1, i_2, ..., i_N\}$, we have that for any $\boldsymbol{\mu} \in \mathcal{R}$, $\varphi(\boldsymbol{\mu}) = \max\{\mathbf{F}\boldsymbol{\mu} - \mathbf{h}\}$ with $\mathbf{F} = [\mathbf{f}_{i_1}, \mathbf{f}_{i_2}, ..., \mathbf{f}_{i_N}]^{\mathrm{T}}$ and $\mathbf{h} = [h_{i_1}, h_{i_2}, ..., h_{i_N}]^{\mathrm{T}}$.

Algorithm 8 details the proposed procedure to learn the classifier parameters $\boldsymbol{\mu}_t$ together with the minimax risk $R(\mathcal{U}_t)$ that provides the performance guarantees. Such algorithm has computational complexity $O(NKm+m^2)$ and memory complexity $O(Nm)$ where $m = |\mathcal{Y}|d$ is the length of the feature mapping, N is the number of subgradients, and K is the number of iterations for ASM in (2.12). Note that the complexity of Algorithms 7 and 8 does not depend on the number of instance-label pairs and time steps. Therefore, the proposed algorithms are applicable to large-scale datasets and have constant complexity per time step.

2.5 Numerical results

This section evaluates the performance of AMRCs in comparison with the presented performance guarantees and the state-of-the-art. In the first set of numerical results, we use a synthetic dataset, while in the second set and the third set of numerical results, we use multiple benchmark datasets. The implementation of the proposed AMRCs is publicly available in Python and Matlab languages.[1] In addition, the appendices provide the detailed description of the benchmark datasets, additional implementation details, and supplementary numerical results.

[1] `https://github.com/MachineLearningBCAM/AMRC-for-concept-drift-ICML-2022`

Algorithm 8 Optimization for AMRC params. and minimax risk

Input: $\boldsymbol{\tau}_t$, $\boldsymbol{\lambda}_t$, $\boldsymbol{\mu}_{t-1}$, x_{t-1}, $\mathbf{F}$, and $\mathbf{h}$
Output: $\boldsymbol{\mu}_t, R(\mathcal{U}_t)$, $\mathbf{F}_{\text{new}}$, and $\mathbf{h}_{\text{new}}$
for $\mathcal{C} \subseteq \mathcal{Y}, \mathcal{C} \neq \emptyset$ **do**
  $\mathbf{F}, \mathbf{h} \leftarrow$ append rows $\sum_{y \in \mathcal{C}} \Phi(x_{t-1}, y)^{\mathrm{T}}/|\mathcal{C}|, 1/|\mathcal{C}|$ to $\mathbf{F}, \mathbf{h}$
$\boldsymbol{\mu}_t^{(1)} \leftarrow \boldsymbol{\mu}_{t-1}$, $\bar{\boldsymbol{\mu}}_t^{(1)} \leftarrow \boldsymbol{\mu}_{t-1}$
for $l = 1, 2, ..., K$ **do**
  $a_l \leftarrow 1/(l+1)^{3/2}$, $\theta_l \leftarrow 2/(l+1)$, $\theta_{l+1} \leftarrow 2/(l+2)$
  $\mathbf{f}_i^{\mathrm{T}} \leftarrow$ row of $\mathbf{F}$ such that $\mathbf{f}_i^{\mathrm{T}} \boldsymbol{\mu}_t^{(l)} - h_i = \max\{\mathbf{F}\boldsymbol{\mu}_t^{(l)} - \mathbf{h}\}$
  $\bar{\boldsymbol{\mu}}_t^{(l+1)} \leftarrow \boldsymbol{\mu}_t^{(l)} + a_l \left(\boldsymbol{\tau}_t - \mathbf{f}_i - \boldsymbol{\lambda}_t \odot \operatorname{sign}(\boldsymbol{\mu}_t^{(l)})\right)$
  $\boldsymbol{\mu}_t^{(l+1)} \leftarrow \bar{\boldsymbol{\mu}}_t^{(l+1)} + \theta_{l+1}(\theta_l^{-1} - 1)\left(\boldsymbol{\mu}_t^{(l)} - \bar{\boldsymbol{\mu}}_t^{(l)}\right)$
  $\mathbf{F}_{\text{new}}, \mathbf{h}_{\text{new}} \leftarrow$ append rows $\mathbf{f}_i^{\mathrm{T}}, h_i$ to $\mathbf{F}_{\text{new}}, \mathbf{h}_{\text{new}}$
$\boldsymbol{\mu}_t \leftarrow \boldsymbol{\mu}_t^{(K+1)}$
$R(\mathcal{U}_t) \leftarrow 1 - \boldsymbol{\tau}_t^{\mathrm{T}} \boldsymbol{\mu}_t + \max\{\mathbf{F}\boldsymbol{\mu}_t - \mathbf{h}\} + \boldsymbol{\lambda}_t^{\mathrm{T}} |\boldsymbol{\mu}_t|$
$\mathbf{F}_{\text{new}}, \mathbf{h}_{\text{new}} \leftarrow$ take the N most recent $\mathbf{f}_i$, h_i from $\mathbf{F}_{\text{new}}$, $\mathbf{h}_{\text{new}}$

We utilize a type of synthetic dataset that has been often used as benchmark for supervised classification under concept drift [113, 114]. At each time t, we generate a label $y_t \in \{1, 2\}$ according to a Bernoulli distribution with parameter $1/2$ and an instance given by

$$\mathbf{x}_t = \Big[4\cos\Big(\pi\Big((\cos(\omega t) - 3)/2 + y_t\Big)\Big) + \epsilon_1, \\ 4\sin\Big(\pi\Big((\cos(\omega t) - 3)/2 + y_t\Big)\Big) + \epsilon_2\Big]^{\mathrm{T}}$$

with $\epsilon_1, \epsilon_2 \sim N(0, 2)$ and $\omega = 0.1$. We use a sinusoidal argument for cosines and sines so that the dataset is even more challenging. The class-conditional underlying distributions $\mathrm{p}_t(\mathbf{x}_t|y_t = 1)$ and $\mathrm{p}_t(\mathbf{x}_t|y_t = 2)$ are Gaussian with means that move with varying velocity and direction in a circle centered at the origin. Specifically, the speed of such means has periodicity π/ω with maxima at times $t = ((n-1)\pi + \pi/2)/\omega$ and minima at times $t = n\pi/\omega$, for $n \in \mathbb{N}$. In addition, the direction of movement changes with the same periodicity at times when the velocity is minimum.

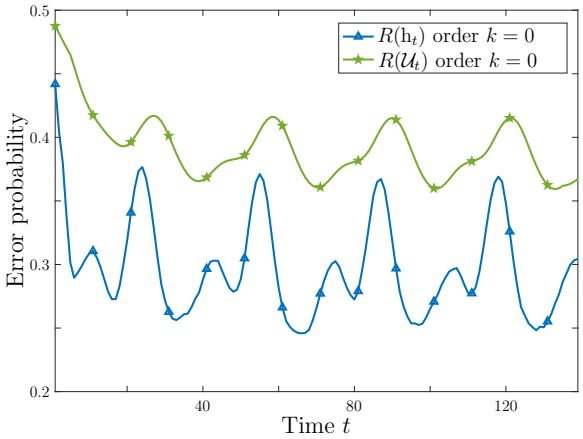
AMRCs are compared with the state-of-the-art using 12 datasets that have been often used as benchmarks for supervised classification under concept drift [63, 74, 75, 107, 113]: "Weather", "Elec2", "Airlines", "German", "Chess", "Usenet1", "Usenet2", "Email Spam", "Credit card", "Smart grid stability", "Shuttle", and "Poker". The last 2 datasets are multi-class problems and the rest are binary. These datasets are further described in Table 2.1 that shows the number of instance-label pairs, the dimensionality of instances, and the number of labels. Three of such datasets are large-scale ("Airlines", "Credit card", and "Poker"), four of them are medium-sized datasets ("Weather",

Table 2.1. Datasets information show number of instances, dimensionality of instances, and number of labels.

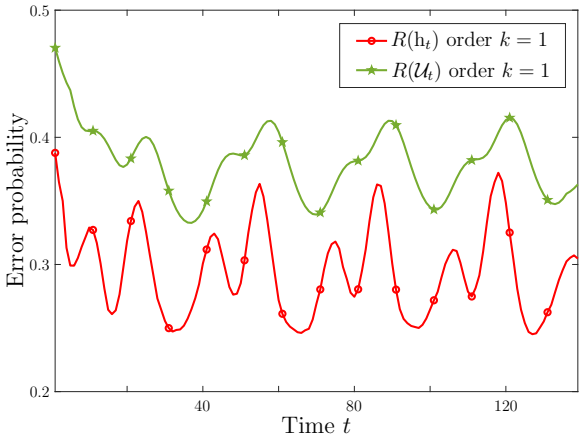
Dataset	Time Steps	Dim. of instances	$\lvert\mathcal{Y}\rvert$
Weather	18,159	8	2
Elec2	1,148	4	2
Airlines	539,383	7	2
German	1,000	24	2
Chess	503	8	2
Usenet 1	1,500	99	2
Usenet 2	1,500	99	2
Email	1,498	913	2
C. card	284,806	30	2
S. Grid	60,000	13	2
Shuttle	100,001	9	4
Poker	829,201	10	10

"Phishing", "Smart grid stability", and "Shuttle"), and the rest are short-sized datasets ("Elec2", "German", "Chess", "Usenet1", "Usenet2", and "Email"). For instance, the "Airlines" dataset used contains 539,383 instances with flight arrival and departure information and aims to predict if a flight will be delayed or not; the "Weather" dataset used contains 18,159 instances with daily measurements of weather factors and aims to predict if it will rain or not; and the "Elec2" dataset used contains 1,148 instances with twice-daily measurements (12am and 12pm) of factors that affect load demand and price and aims to predict if the price will be higher or not. The benchmark datasets can be obtained from UCI repository and from the Massive On-line Analysis library [93].

The results for AMRCs are obtained using the feature mapping described in equation (1) of Section 1.2. Specifically, for the synthetic dataset we use the linear map $\Psi(x) = x$ and for the benchmark datasets we use random Fourier featuress (RFFs) with $D = 200$ as given by (**??**). The scaling parameter γ is calculated using a two-stage five-fold cross validation. Specifically, at the first stage, the values for the scaling factor are selected from 2^i for $i = \{-6, -3, 0, 3, 6\}$. At the second stage, if $\gamma_0 = 2^i$ where i is the best parameter obtained at the first stage, then the values for the scaling parameters are selected from $\gamma_0 2^i$, $i = \{-2, -1, 0, 1, 2\}$. The final value is $\gamma = \gamma_0 2^i$ where i is the best parameter obtained at second stage. In addition, we use the recursive approach presented in [111] to obtain the variances $\mathbf{Q}_{t,i}$ and $r^2_{t,i}$ of noise processes $\mathbf{w}_{t,i}$ and $v_{t,i}$, in (2.6); we obtain the probability of the labels using (2.5) with $W = 200$; and the ASM in (2.12) is implemented with $N = 100$ and $K = 2000$. AMRCs are compared with 10 state-of-the-art techniques: AdaRaker [48], randomized budget perceptron (RBP) [61], Projectron, Projectron ++ [46], naive online regularized risk mini-



(a). Instantaneous bounds and probabilities of error for AMRC of order $k = 0$.



(b). Instantaneous bounds and probabilities of error for AMRC of order $k = 1$.

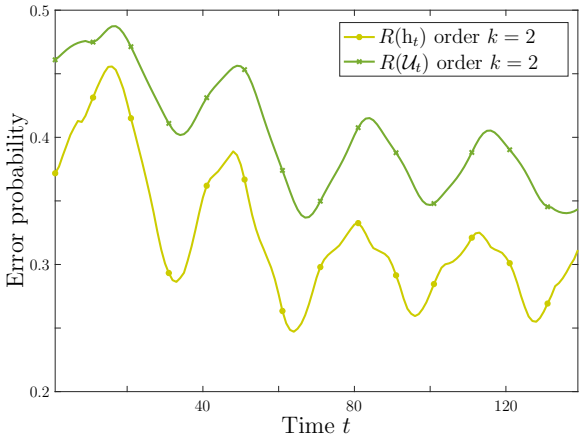
Figure 2.3. Results on synthetic data show the evolution in time of instantaneous bounds and probabilities of error.

mization algorithm (NORMA) [47], Nyström online gradient descent (NOGD), Fourier online gradient descent (FOGD) [74], λ - perceptron [63], dynamic weighted majority (DWM) [91], and Forgetron [94]. The results of the 9 methods that use kernels utilize a scaling parameter calculated with a two-stage five-fold cross validation.

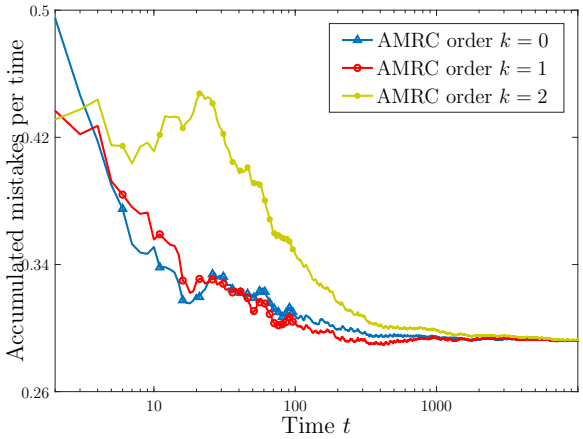
In the first set of numerical results we show the reliability of the presented instantaneous bounds using the synthetic data since the error probability in each time step cannot be computed using real-world datasets. In order to quantify the error probability at each time, we use 10,000 Monte Carlo simulations. Figures 2.3 and 2.4a show the averaged instantaneous bounds of error probabilities corresponding to first inequality in (2.3) for $\alpha_t = 0$ in comparison with the true error probabilities $R(\mathrm{h}_t)$ at each time. Such figure shows that the instantaneous bounds given by $R(\mathcal{U}_t)$ can offer tight upper bounds for the error probability at each time. In addition, Figure 2.4b shows the accumulated mistakes per time step of AMRCs of order k = 0, 1, and 2. As can be seen in such figure, an increased order can result in an improved overall performance at the expenses of worse initial performance.

In the second set of numerical results we use benchmark datasets to quantify AMRCs performance with respect to the state-of-the-art, the improvement due to multidimensional adaptation, and the reliability of the presented mistake bounds. Table 2.2 shows the classification error of AMRCs of order $k = 1$ and the 10 state-of-the-art techniques for the 12 benchmark datasets described above.[2] The table shows that AMRCs and deterministic AMRCs (Det. AMRC) achieve the best performance in 3 and 9 datasets, respectively, and are competitive in all datasets. In addition, Table 2.3 shows the averaged running time in milliseconds of AMRCs and the 10 state-of-the-art techniques

[2]Bold numbers indicate the top result and N/As indicates that the classification method cannot be used with non-binary datasets.



(a). Instantaneous bounds and probabilities of error for AMRC of order $k = 2$.



(b). Results on synthetic data show the evolution in time of accumulated mistakes per number of steps.

for the 12 benchmark datasets. Such table shows the running time of AMRCs using $K = 500$ and $K = 2000$ iterations in the ASM in Section 2.4, and Table 2.4 shows the classification error of AMRCs with $K = 500$ and $K = 2000$. The running time of AMRCs is of the order of tens of milliseconds per time step similarly to state-of-the-art techniques.

Table 2.2 shows that AMRCs offer an overall improved performance compared to the state-of-the-art along the benchmark datasets. In order to more clearly assess the performance improvement due to the multidimensional adaptation, we also implement a simple version of deterministic AMRCs that only account for a scalar rate of change (Unidim. AMRCs). Such AMRCs are obtained by using, for all the components $i = 1, 2, ..., m$, the same gain in (2.7) and (2.8) that is taken as the average of the gains in (2.9). As shown in the table, the presented multidimensional adaptation can enable significant performance improvements in most datasets.

Figures 2.5a and 2.5b provide more detailed comparisons using the 3 techniques that have the top average rank (Det. AMRC, AMRC, and DWM) in "German" and "Usenet1" datasets. In addition, Figures 2.5a and 2.5b show the accumulated mistakes per number of steps $\sum_{t=1}^{T} \mathbb{1}\{\hat{y}_t \neq y_t\}/T$ at each time in comparison with the accumulated mistake bounds per number of steps corresponding to the first inequality in (2.4) for $\delta = 0.05$ and $\alpha_t = 0$. As can be seen in such figure, the accumulated mistake bounds in (2.4) obtained at AMRC learning can offer accurate estimates for the classification error. These results together with those in Figure 2.3b show that the proposed methods can provide tight performance guarantees both in terms of instantaneous bounds for error probabilities and also in terms of bounds for accumulated mistakes.

Table 2.2. Classification error in % of AMRCs in comparison with state-of-the-art techniques.

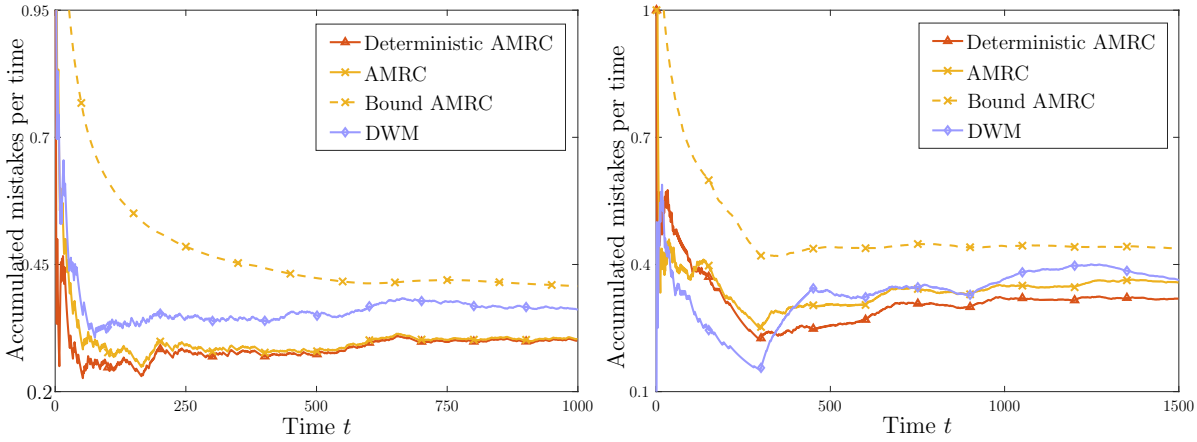
Algorithm	Weather	Elec2	Airlines	German	Chess	Usenet1	Usenet2	Email	C. card	S. Grid	Shuttle	Poker	Ave. rank
AdaRaker	34.8	49.5	42.7	41.0	46.6	50.0	49.0	45.7	0.33	46.5	N/A	N/A	10.9
RBP	34.3	39.6	42.7	40.5	36.5	44.9	48.3	45.3	0.33	38.6	N/A	N/A	8.0
Projectron	30.7	36.8	41.8	40.0	35.1	49.1	47.7	48.2	0.33	**34.4**	64.7	22.6	6.2
Projectron ++	32.0	48.1	41.9	42.2	43.4	47.2	45.2	48.1	0.33	35.8	64.9	23.4	8.3
NORMA	35.5	43.2	40.7	39.7	39.4	36.0	32.8	34.6	0.18	38.8	N/A	N/A	6.6
NOGD	38.2	48.2	39.1	37.1	39.0	34.6	33.4	36.1	0.44	38.7	69.3	23.4	7.4
FOGD	32.0	44.9	41.1	37.6	38.3	45.9	47.0	44.9	0.33	36.0	64.9	22.9	6.4
λ-perceptron	31.4	40.4	44.5	42.9	41.6	44.5	31.4	46.8	0.18	35.8	N/A	N/A	7.3
DWM	**30.0**	36.3	**37.8**	41.2	35.2	36.3	**28.8**	39.5	0.80	36.3	16.9	22.0	4.8
Forgetron	32.0	46.0	42.0	40.1	33.1	46.8	47.2	41.7	0.33	45.5	64.4	25.6	7.3
Unidim. AMRC	31.3	40.1	44.5	30.3	38.3	46.3	33.3	48.4	0.18	36.2	70.4	39.4	7.2
AMRC	32.3	35.8	38.9	30.3	**27.7**	35.7	30.9	43.7	**0.17**	35.8	15.2	**21.9**	3.0
Det. AMRC	**30.0**	**33.9**	39.4	**30.0**	33.4	**32.0**	29.9	**33.9**	**0.17**	**34.4**	**10.6**	**21.9**	**1.5**

Table 2.3. Averaged running times per time step in milliseconds of AMRC in comparison with state-of-the-art techniques.

Algorithm	Mean
RBP	9.2
Proj.	9.0
P. ++	9.1
NORMA	6.8
A.Raker	10.1
NOGD	4.6
FOGD	1.9
λ-per.	12.2
DWM	6.6
Forg.	6.9
AMRC $K = 500$	14.4
AMRC $K = 2000$	51.9

Table 2.4. Classification error in % of AMRC with $K = 500$ and $K = 2000$ iterations in ASM.

Dataset	AMRC	
	$K = 500$	$K = 2000$
Weather	32.4	32.3
Elec2	35.9	35.8
Airlines	39.2	38.9
German	30.4	30.3
Chess	28.8	27.7
Usenet1	36.8	35.7
Usenet2	30.6	30.9
Email	43.0	43.7
C. card	2.53	0.17
S. grid	36.3	35.8
Shuttle	44.6	15.2
Poker	30.4	21.9



(a) Results on "German" dataset shows the evolution of accumulated mistake bounds and accumulated mistakes per number of steps.

(b). Results on "Usenet1" dataset shows the evolution of accumulated mistake bounds and accumulated mistakes per number of steps.

Figure 2.5: Results on real-world datas shows the evolution of accumulated mistake bounds.

2.6 Conclusion

The chapter proposes the methodology of adaptive minimax risk classifiers (AMRCs) with multidimensional adaptation and performance guarantees. AMRCs learn classification rules estimating multiple statistical characteristics of the time-varying underlying distribution. In addition, AMRCs provide both qualitative and computable tight performance guarantees. The chapter also proposes algorithms to track the time-varying underlying distribution accounting for multivariate and high-order time changes, and algorithms to efficiently update classification rules as new instance-label pairs arrive. The numerical results assess the performance improvement of AMRCs with respect to the state-of-the-art using benchmark datasets. The proposed methodology enables to more fully exploit data with characteristics that change over time and can provide more informative performance guarantees under concept drift.

Chapter 3

Minimax Classification for Continual Learning

3.1 Introduction

In practical scenarios, classification problems (tasks) often have limited sample sizes and arrive sequentially over time. Continual learning (CL) (also known as lifelong learning) can boost the effective sample size (ESS) of each task by leveraging information from preceding and succeeding tasks (forward and backward learning) [7, 8, 11]. The general goal of such approaches is to replicate the humans' ability to continually improve the performance of each task exploiting information acquired from other tasks.

The development of CL techniques is hindered by the continuous arrival of samples from tasks characterized by different underlying distributions. In particular, backward learning (also known as reverse transfer) is often prone to a so-called catastrophic forgetting in which a task's performance gets worse while trying to repeatedly incorporate information from the succeeding tasks [14–16]. More generally, CL methods face a so-called stability-plasticity dilemma: the excessive usage of information from different tasks can result in a performance decrease while a moderate usage does not fully exploit the potential of CL [115, 116].

Most of CL techniques are designed for tasks sampled i.i.d. from a task environment [50, 117, 118], and current methods cannot capture the usual higher similarities between consecutive tasks. For a sequence of tasks that arrive over time, it is common that the tasks are time-dependent and consecutive tasks are significantly more similar. For instance, if each task corresponds to the classification of portraits from a specific time period [19], the similarity between tasks is markedly higher for consecutive tasks (see Figure 3.1). In the current literature of CL, only [64] considers scenarios with time-dependent tasks and analyzes the feasibility of transferring information from the preceding tasks. On the other hand, methods designed for concept drift adaptation [6, 104, 119] account for time-dependent underlying distributions but only aim to learn the last task in the sequence.

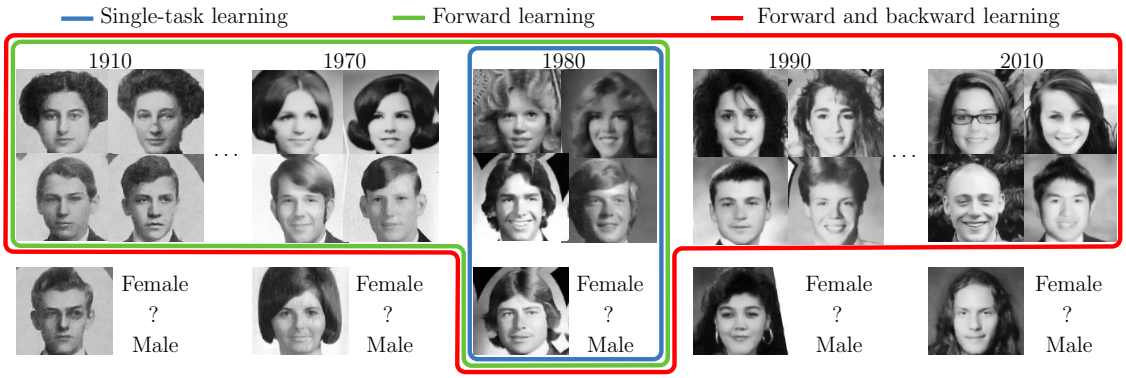


Figure 3.1: For tasks that arrive over time, consecutive tasks are often more similar. Forward and backward learning can exploit such similarities and extract information from preceding and succeding tasks.

This chapter presents continual learning methods based on minimax risk classifiers (CL-MRCs). The proposed techniques effectively exploit forward and backward learning and account for time-dependent tasks. Specifically, the main contributions presented in the chapter are as follows.

- The presented CL-MRCs minimize the worst-case error probabilities over uncertainty sets obtained using information from all the tasks.
- We propose learning techniques that can effectively incorporate information from the ever-increasing sequence of tasks and provide performance guarantees for forward and backward learning.
- We analytically characterize the increase in ESS provided by forward and backward learning in terms of the expected quadratic change between consecutive tasks.
- We numerically quantify the performance improvement provided by the presented learning techniques in comparison with existing methods using multiple datasets, different sample sizes, and number of tasks.

Notations Calligraphic letters represent sets; $\|\cdot\|_1$ and $\|\cdot\|_\infty$ denote the 1-norm and the infinity norm of its argument, respectively; $\preceq$ and $\succeq$ denote vector inequalities; $\mathbb{I}\{\cdot\}$ denotes the indicator function; and $\mathbb{E}_{\mathrm{p}}\{\cdot\}$ and $\mathbb{V}\mathrm{ar}_{\mathrm{p}}\{\cdot\}$ denote the expectation and the variance of its argument with respect to distribution p. For a vector $\mathbf{v}$, $v^{(i)}$ and $\mathbf{v}^{\mathrm{T}}$ denote the i-th component and the transpose of $\mathbf{v}$. Non-linear operators acting on vectors denote component-wise operations. For instance, $|\mathbf{v}|$ and $\mathbf{v}^2$ denote the vector formed by the absolute value and the square of each component, respectively.

3.2 Problem formulation

In CL, sample sets $D_1, D_2, \ldots$ arrive over time steps $1, 2, \ldots$ corresponding with different classification tasks characterized by underlying distributions $\mathrm{p}_1, \mathrm{p}_2, \ldots$. At each time step k, CL methods aim to obtain classification rules $\mathrm{h}_1, \mathrm{h}_2, \ldots, \mathrm{h}_k$ with small expected losses $\ell(\mathrm{h}_1, \mathrm{p}_1), \ell(\mathrm{h}_2, \mathrm{p}_2), \ldots, \ell(\mathrm{h}_k, \mathrm{p}_k)$ for the current sequence of k tasks. For instance, overall performance is usually assessed by the averaged error $\frac{1}{k}\sum_{i=1}^{k} \ell(\mathrm{h}_i, \mathrm{p}_i)$. As depicted in Figure 3.1, for each j-th task with $j \in \{1, 2, \ldots, k\}$, CL methods obtain the classification rule h_j leveraging information obtained from sample sets $D_1, D_2, \ldots, D_j$ (forward learning) and from sample sets $D_{j+1}, D_{j+2}, \ldots, D_k$ (backward learning).

As described in Section 1.2, most existing CL techniques are designed for tasks characterized by distributions $\mathrm{p}_1, \mathrm{p}_2, \ldots$ such that the tasks' distributions satisfy (i.i.d.-A). In the following, we propose CL techniques designed for time-dependent tasks that are characterized by distributions $\mathrm{p}_1, \mathrm{p}_2, \ldots$ such that the tasks' distributions satisfy (TD-A).

3.3 Forward learning with performance guarantees

This section presents the recursions that allow to obtain mean and mean squared error (MSE) vectors for each task retaining information from preceding tasks. In addition, it characterizes the increase in ESS provided by forward learning in terms of the tasks' expected quadratic change and the number of tasks.

3.3.1 Forward learning

The proposed techniques for forward learning account for time-dependent tasks and obtain classification rules for each task leveraging information from preceding tasks. Let $\boldsymbol{\tau}_j^{\rightarrow}$ and $\boldsymbol{s}_j^{\rightarrow}$ denote the mean and MSE vectors for forward learning corresponding to the j-th task for $j \in \{1, 2, \ldots, k\}$. The following recursions allow to obtain $\boldsymbol{\tau}_j^{\rightarrow}$ and $\boldsymbol{s}_j^{\rightarrow}$ for each j-th task using those vectors for the preceding task $\boldsymbol{\tau}_{j-1}^{\rightarrow}, \boldsymbol{s}_{j-1}^{\rightarrow}$ as

$$\boldsymbol{\tau}_j^{\rightarrow} = \boldsymbol{\tau}_j + \frac{\boldsymbol{s}_j}{\boldsymbol{s}_{j-1}^{\rightarrow} + \boldsymbol{s}_j + \boldsymbol{d}_j^2} \left(\boldsymbol{\tau}_{j-1}^{\rightarrow} - \boldsymbol{\tau}_j\right) \tag{3.1}$$

$$\boldsymbol{s}_j^{\rightarrow} = \left(\frac{1}{\boldsymbol{s}_j} + \frac{1}{\boldsymbol{s}_{j-1}^{\rightarrow} + \boldsymbol{d}_j^2}\right)^{-1} \tag{3.2}$$

with $\boldsymbol{\tau}_j$ and $\boldsymbol{s}_j$ given by (7) and $\boldsymbol{\tau}_1^{\rightarrow} = \boldsymbol{\tau}_1$ and $\boldsymbol{s}_1^{\rightarrow} = \boldsymbol{s}_1$.

The vector $\boldsymbol{d}_j^2$ assesses the expected quadratic change between consecutive tasks. In the following, the change between consecutive tasks is described by $\boldsymbol{w}_j = \boldsymbol{\tau}_j^{\infty} - \boldsymbol{\tau}_{j-1}^{\infty}$ for any $j \in \{2, 3, \ldots, k\}$, where $\boldsymbol{\tau}_j^{\infty} = \mathbb{E}_{\mathrm{p}_j}\{\Phi(x, y)\}$ is the expectation of the feature mapping with respect to the underlying distribution. If $\mathrm{p}_j - \mathrm{p}_{j-1}$ are independent and

zero-mean for $j = 2, 3, \ldots$, then vectors $\boldsymbol{w}_j$ are also independent and zero-mean for any feature mapping.

Taking $\boldsymbol{d}_i^2 = \mathbb{E}\{\boldsymbol{w}_i^2\} = \mathbb{E}\{(\boldsymbol{\tau}_i^\infty - \boldsymbol{\tau}_{i-1}^\infty)^2\}$ and $\boldsymbol{\sigma}_i^2 = \mathbb{V}\mathrm{ar}_{\mathrm{p}_i}\{\Phi(x, y)\}$ for any i, the recursion in (3.1) provides the unbiased linear estimator of the mean vector $\boldsymbol{\tau}_j^\infty$ based on $D_1, D_2, \ldots, D_j$ that has the minimum MSE, while the recursion in (3.2) provides its MSE (see Appendix A.1 for a detailed derivation). Vectors $\boldsymbol{\sigma}_i^2$ and $\boldsymbol{d}_i^2$ can be estimated online using the sample sets. In particular, $\boldsymbol{\sigma}_i^2$ can be estimated as the sample variance, while $\boldsymbol{d}_i^2$ can be estimated using sample averages as

$$\boldsymbol{d}_i^2 = \frac{1}{W} \sum_{l=1}^{W} (\boldsymbol{\tau}_{i_l} - \boldsymbol{\tau}_{i_{l-1}})^2 \tag{3.3}$$

where $i_0, i_1, \ldots, i_W$ are the $W + 1$ closest indexes to i in $\{1, 2, \ldots, k\}$.

Recursions (3.1)-(3.2) obtain mean and MSE vectors for the j-th task by acquiring information from the j-th sample set D_j and retaining information from preceding tasks. Specifically, recursion (3.1) obtains the mean vector $\boldsymbol{\tau}_j^{\rightarrow}$ by adding a correction to the sample average $\boldsymbol{\tau}_j$. This correction is proportional to the difference between $\boldsymbol{\tau}_j$ and $\boldsymbol{\tau}_{j-1}^{\rightarrow}$ with a proportionality constant that depends on the MSE vectors $\boldsymbol{s}_j, \boldsymbol{s}_{j-1}^{\rightarrow}$ and the expected quadratic change $\boldsymbol{d}_j^2$. In particular, if $\boldsymbol{s}_j \ll \boldsymbol{s}_{j-1}^{\rightarrow} + \boldsymbol{d}_j^2$, the mean vector is given by the sample average as in single-task learning, and if $\boldsymbol{s}_j \gg \boldsymbol{s}_{j-1}^{\rightarrow} + \boldsymbol{d}_j^2$, the mean vector is given by that of the preceding task. Note that for forward learning, at each step k, only the vectors for the last task $\boldsymbol{\tau}_k^{\rightarrow}$ and $\boldsymbol{s}_k^{\rightarrow}$ need to be obtained from those of the $(k-1)$-th task. The vectors for the remaining j-th tasks with $j \in \{1, 2, \ldots, k-1\}$ stay the same as at step $k - 1$ (see also Fig. 3.2 and Alg. 9 below).

3.3.2 Performance guarantees and effective sample sizes

The following result provides bounds for the minimax risk for each task with respect to the smallest minimax risk. For each j-th task, we denote by R_j^∞ the smallest minimax risk and by $\boldsymbol{\mu}_j^\infty$ the classifier parameter that determines the optimal minimax rule, as described in Section 0.4. In addition, we denote by $R(\mathcal{U}_j^{\rightarrow})$ the minimax risk over uncertainty set $\mathcal{U}_j^{\rightarrow}$ determined as in (2) using the mean and confidence vectors $\boldsymbol{\tau}_j^{\rightarrow}$ and $\boldsymbol{\lambda}_j^{\rightarrow} = \sqrt{\boldsymbol{s}_j^{\rightarrow}}$ provided by (3.1) and (3.2).

Theorem 8. Let M and κ be such that $M \geq \|\Phi(x, y)\|_\infty \; \forall (x, y) \in \mathcal{X} \times \mathcal{Y}$ and

$$\kappa \geq \frac{\sigma\left(\Phi_j^{(i)}\right)}{\sigma_j^{(i)}}, \quad \kappa \geq \frac{\sigma\left(w_j^{(i)}\right)}{d_j^{(i)}}$$

for $j = 1, 2, \ldots, k$ and $i = 1, 2, \ldots, m$, where $\Phi_j^{(i)}$ denotes the r.v. given by the i-th component of the feature mapping of samples from the j-th task, and $\sigma(z)$ denotes the sub-Gaussian parameter of a r.v. z. For any $j \in \{1, 2, \ldots, k\}$, with probability at least

$1-\delta$ we have that

$$R(\mathcal{U}_j^{\rightarrow}) \leq R_j^{\infty} + \frac{M(\kappa+1)\sqrt{2\log(2m/\delta)}}{\sqrt{n_j^{\rightarrow}}} \left\|\boldsymbol{\mu}_j^{\infty}\right\|_1 \tag{3.4}$$

with $n_1^{\rightarrow} = n_1$ and $n_j^{\rightarrow} \geq n_j + n_{j-1}^{\rightarrow} \frac{\|\boldsymbol{\sigma}_j^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty + n_{j-1}^{\rightarrow}\|\boldsymbol{d}_j^2\|_\infty}$ for $j \geq 2$.

Proof. See Appendix A.4. □

The excess risk in inequality (3.4) decreases as $\mathcal{O}(1/\sqrt{n_j^{\rightarrow}})$ using the forward learning methods proposed, while such difference would decrease as $\mathcal{O}(1/\sqrt{n_j})$ using only the information of the j-th task. Therefore, $n_j^{\rightarrow}$ in (3.4) is the ESS of the proposed CL-MRC method with forward learning. The ESS of each task is obtained by adding a fraction of the ESS for the preceding task to the sample size. In particular, if $\boldsymbol{d}_j^2$ is large, the ESS is given by the sample size, while if $\boldsymbol{d}_j^2$ is small, the ESS is given by the sum of the sample size and the ESS of the preceding task.

Other existing methods provide comparable performance bounds [64, 72]. Such bounds decrease with the number of tasks and increase with the change between consecutive distributions. Specifically, bounds in Proposition 1 of [72] and in Theorem 7 of [64] are proportional to the discrepancy and to the Kullback-Leibler divergence between consecutive distributions. The bound in Theorem 8 above decreases with the number of tasks and increases with the expected quadratic change $\boldsymbol{d}_j^2$ between consecutive distributions. Note that the coefficient κ in (3.4) can be taken to be small as long as the values used for σ_j and d_j are not much lower than the sub-Gaussian parameters of Φ_j and w_j, respectively. In particular, κ is smaller than the maximum of $M/\min_{j,i}\{\sigma_j^{(i)}\}$ and $2M/\min_{j,i}\{d_j^{(i)}\}$ due to the bound for the sub-Gaussian parameter of bounded random variables (see e.g., Section 2.1.2 in [85]).

Theorem 8 shows the increase in ESS in terms of the ESS of the preceding task. The following result allows to directly quantify the ESS in terms of the sample size and the expected quadratic change.

Theorem 9. Let d, $\boldsymbol{\sigma}_j$ and n be such that $d^2 \geq \|\boldsymbol{d}_j^2\|_\infty$, $\|\boldsymbol{\sigma}_j^2\|_\infty \leq 1$, and $n \leq n_j$ for $j = 1, 2, \ldots, k$. For any $j \in \{1, 2, \ldots, k\}$, we have that the ESS in (3.4) can be taken so that it satisfies

$$n_j^{\rightarrow} \geq n\left(1 + \frac{(1+\alpha)^{2j-1} - 1 - \alpha}{\alpha(1+\alpha)^{2j-1} + \alpha}\right) \tag{3.5}$$

with $\alpha = \frac{2}{\sqrt{1+\frac{4}{nd^2}}-1}$. In particular, for $j \geq 2$, we have that

$$\begin{aligned}
n_j^{\rightarrow} &\geq n\left(1 + \frac{j-1}{3}\right) && \text{if} && nd^2 < \frac{1}{j^2} \\
n_j^{\rightarrow} &\geq n\left(1 + \frac{1}{5\sqrt{nd^2}}\right) && \text{if} && \frac{1}{j^2} \leq nd^2 < 1 \\
n_j^{\rightarrow} &\geq n\left(1 + \frac{1}{3nd^2}\right) && \text{if} && nd^2 \geq 1.
\end{aligned}$$

Proof. See Appendix A.5. □

The above theorem characterizes the increase in ESS provided by forward learning in terms of the tasks' expected quadratic change. Such increase grows monotonically with the number of preceding tasks j as shown in (3.5) and becomes proportional to j when the expected quadratic change is smaller than $1/(j^2 n)$. Figure 3.3 below further illustrates the increase in ESS with respect to the sample size $(n_j^{\rightarrow}/n)$ due forward learning in comparison with forward and backward learning.

3.4 Forward and backward learning with performance guarantees

This section presents the recursions that allow to obtain mean and MSE vectors for each task retaining information from preceding tasks and acquiring information from succeeding tasks. In addition, it characterizes the increase in ESS provided by forward and backward learning in terms of the tasks' expected quadratic change and the number of tasks.

Backward learning is more challenging than forward learning since, for each task, the sequence of succeeding tasks is ever-increasing due to the continuous arrival of tasks, while the sequence of preceding tasks is always the same. The repeated usage of information from the succeeding tasks can result in a so-called catastrophic forgetting in which the tasks' performance gets worse over time. The techniques proposed below for backward learning effectively increase the ESS over time by carefully accounting for the new information at each step.

3.4.1 Forward and backward learning

The proposed techniques for forward and backward learning account for time-dependent tasks and obtain classification rules for each task leveraging information from preceding and succeeding tasks. From preceding tasks, we obtain the forward mean and MSE vectors $\boldsymbol{\tau}_j^{\rightarrow}, \boldsymbol{s}_j^{\rightarrow}$ using recursions (3.1)-(3.2), while from succeeding tasks, we obtain the backward mean and MSE vectors $\boldsymbol{\tau}_j^{\leftarrow k}, \boldsymbol{s}_j^{\leftarrow k}$ using recursions (3.1)-(3.2) in retrodiction. Specifically, vectors $\boldsymbol{\tau}_j^{\leftarrow k}$ and $\boldsymbol{s}_j^{\leftarrow k}$ are obtained using the same recursion as for $\boldsymbol{\tau}_j^{\rightarrow}$ and $\boldsymbol{s}_j^{\rightarrow}$ in (3.1)-(3.2) with $\boldsymbol{s}_{j+1}^{\leftarrow k}, \boldsymbol{d}_{j+1}^2$, and $\boldsymbol{\tau}_{j+1}^{\leftarrow k}$ instead of $\boldsymbol{s}_{j-1}^{\rightarrow}, \boldsymbol{d}_j^2$, and $\boldsymbol{\tau}_{j-1}^{\rightarrow}$.

Let $\boldsymbol{\tau}_j^{\rightleftharpoons k}$ and $\boldsymbol{s}_j^{\rightleftharpoons k}$ denote the mean and MSE vectors for forward and backward learning corresponding to the j-th task for $j \in \{1, 2, \ldots, k\}$. The following recursions allow to obtain, at each step k, the mean and MSE vectors $\boldsymbol{\tau}_j^{\rightleftharpoons k}$ and $\boldsymbol{s}_j^{\rightleftharpoons k}$ for each j-th task using those vectors for forward learning $\boldsymbol{\tau}_j^{\rightarrow}, \boldsymbol{s}_j^{\rightarrow}$ and backward learning $\boldsymbol{\tau}_{j+1}^{\leftarrow k}, \boldsymbol{s}_{j+1}^{\leftarrow k}$

as

$$\boldsymbol{\tau}_j^{\rightleftharpoons k} = \boldsymbol{\tau}_j^{\rightarrow} + \frac{\boldsymbol{s}_j^{\rightarrow}}{\boldsymbol{s}_j^{\rightarrow} + \boldsymbol{s}_{j+1}^{\leftarrow k} + \boldsymbol{d}_{j+1}^2} \left(\boldsymbol{\tau}_{j+1}^{\leftarrow k} - \boldsymbol{\tau}_j^{\rightarrow}\right) \tag{3.6}$$

$$\boldsymbol{s}_j^{\rightleftharpoons k} = \left(\frac{1}{\boldsymbol{s}_j^{\rightarrow}} + \frac{1}{\boldsymbol{s}_{j+1}^{\leftarrow k} + \boldsymbol{d}_{j+1}^2}\right)^{-1} \tag{3.7}$$

with $\boldsymbol{\tau}_k^{\leftarrow k} = \boldsymbol{\tau}_k, \boldsymbol{s}_k^{\leftarrow k} = \boldsymbol{s}_k$ and $\boldsymbol{\tau}_k^{\rightleftharpoons k} = \boldsymbol{\tau}_k^{\rightarrow}, \boldsymbol{s}_k^{\rightleftharpoons k} = \boldsymbol{s}_k^{\rightarrow}$. Analogously to the case of forward learning in Section 3.3.1, taking $\boldsymbol{d}_i^2 = \mathbb{E}\{\boldsymbol{w}_i^2\}$ and $\boldsymbol{\sigma}_i^2 = \mathbb{V}\mathrm{ar}_{\mathrm{p}_i}\{\Phi(x, y)\}$ for any i, the recursion in (3.6) provides the unbiased linear estimator of the mean vector $\boldsymbol{\tau}_j^{\infty}$ based on $D_1, D_2, \ldots, D_j$ and $D_{j+1}, D_{j+2}, \ldots, D_k$ that has the minimum MSE, while the recursion in (3.7) provides its MSE (see Appendix A.1 for a detailed derivation).

Recursions (3.6)-(3.7) obtain at step k the mean and MSE vectors for the j-th task by retaining information from preceding tasks and acquiring information from succeeding tasks. Specifically, recursion (3.6) obtains the mean vector $\boldsymbol{\tau}_j^{\rightleftharpoons k}$ by adding a correction to the mean vector of the corresponding task $\boldsymbol{\tau}_j^{\rightarrow}$ obtained for forward learning. This correction is proportional to the difference between $\boldsymbol{\tau}_j^{\rightarrow}$ and $\boldsymbol{\tau}_{j+1}^{\leftarrow k}$ with a proportionality constant that depends on the MSE vectors $\boldsymbol{s}_j^{\rightarrow}, \boldsymbol{s}_{j+1}^{\leftarrow k}$ and the expected quadratic change $\boldsymbol{d}_{j+1}^2$. In particular, if $\boldsymbol{s}_j^{\rightarrow} \ll \boldsymbol{s}_{j+1}^{\leftarrow k} + \boldsymbol{d}_{j+1}^2$, the mean vector is given by that of the corresponding task for forward learning, and if $\boldsymbol{s}_j^{\rightarrow} \gg \boldsymbol{s}_{j+1}^{\leftarrow k} + \boldsymbol{d}_{j+1}^2$, the mean vector is given by that of the succeeding task for backward learning.

3.4.2 Implementation

This section describes the implementation of the proposed CL-MRCs with forward and backward learning and its computational and memory complexities.

Figure 3.2 depicts the flow diagram for the proposed CL-MRC methodology. The proposed techniques carefully avoid the repeated usage of the same information from the sequence of succeeding tasks.

At each step k, new backward mean vectors $\boldsymbol{\tau}_{j+1}^{\leftarrow k}$ for $j = k-1, k-2, .., k-b$ are obtained for b backward steps, then the forward and backward mean vectors $\boldsymbol{\tau}_j^{\rightleftharpoons k}$ given by (3.6) are obtained from the forward mean vectors $\boldsymbol{\tau}_j^{\rightarrow}$ and the backward mean vectors $\boldsymbol{\tau}_{j+1}^{\leftarrow k}$. In particular, $\boldsymbol{\tau}_j^{\rightarrow}$ provides the information from the preceding tasks $1, 2, \ldots, j$, while $\boldsymbol{\tau}_{j+1}^{\leftarrow k}$ provides the information from the succeeding tasks $j+1, j+2, \ldots, k$.

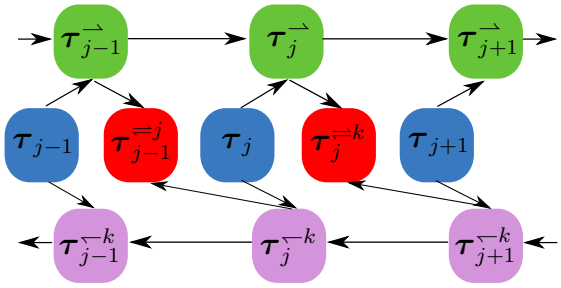


Figure 3.2: Diagram for CL-MRC methodology.

Algorithm 9 details the implementation of the proposed CL-MRCs at each

step. For k steps, CL-MRCs have computational complexity $\mathcal{O}((b+1)Kmk)$ and memory complexity $\mathcal{O}((b+k)m)$ where K is the number of iterations used for the convex optimization problem (6), m is the length of the feature vector, and b is the number of backward steps. In particular, if $b = 0$, CL-MRC carries out only forward learning. The complexity of forward and backward learning increases proportionally to the number of backward steps that can be taken to be rather small, as shown in the following. Even more efficient implementations can be obtained using Rauch-Tung-Striebel recursions (see e.g., Section 7.2 in [120]) that can obtain $\boldsymbol{\tau}_j^{\rightleftharpoons k}$ from $\boldsymbol{\tau}_{j+1}^{\rightleftharpoons k}$ as shown in equations (1.8)-(1.9) in Section 1.4.

Algorithm 9 CL-MRC at step k

Input: D_k from new task and $\boldsymbol{\tau}_j, \boldsymbol{s}_j, \boldsymbol{\tau}_j^{\rightarrow}, \boldsymbol{s}_j^{\rightarrow}$ for $k-b \leq j < k$ from previous $b-1$ steps

Output: $\boldsymbol{\mu}_j$ for $k-b \leq j \leq k$, $\boldsymbol{\tau}_k, \boldsymbol{s}_k, \boldsymbol{\tau}_k^{\rightarrow}, \boldsymbol{s}_k^{\rightarrow}$

Obtain sample average and MSE vectors $\boldsymbol{\tau}_k^{\leftarrow k} = \boldsymbol{\tau}_k, \boldsymbol{s}_k^{\leftarrow k} = \boldsymbol{s}_k$ using the sample set D_k ▷ Single-task

Estimate the tasks' expected quadratic change $\boldsymbol{d}_k^2$ using (3.3)

Obtain the forward mean and MSE vectors $\boldsymbol{\tau}_k^{\rightarrow}, \boldsymbol{s}_k^{\rightarrow}$ using (3.1)-(3.2) ▷ Forward

Take $\boldsymbol{\lambda}_k^{\rightarrow} = \sqrt{\boldsymbol{s}_k^{\rightarrow}}$ and obtain classifier parameter $\boldsymbol{\mu}_k$ solving the optimization problem (6)

for $j = k-1, k-2, \ldots, k-b$ **do**

Estimate the tasks' expected quadratic change $\boldsymbol{d}_j^2$ using (3.3)

Obtain backward mean and MSE vectors $\boldsymbol{\tau}_{j+1}^{\leftarrow k}, \boldsymbol{s}_{j+1}^{\leftarrow k}$ using (3.1)-(3.2) in retrodiction ▷ Backward

Obtain mean and MSE vectors $\boldsymbol{\tau}_j^{\rightleftharpoons k}, \boldsymbol{s}_j^{\rightleftharpoons k}$ using (3.6)-(3.7) ▷ Forward and backward

Take $\boldsymbol{\lambda}_j^{\rightleftharpoons k} = \sqrt{\boldsymbol{s}_j^{\rightleftharpoons k}}$ and obtain classifier parameters $\boldsymbol{\mu}_j$ solving the optimization problem (6)

3.4.3 Performance guarantees and effective sample sizes

The following result provides bounds for the minimax risk for each task with respect to the smallest minimax risk. For each j-th task and step k, we denote by $R(\mathcal{U}_j^{\rightleftharpoons k})$ the minimax risk over uncertainty set $\mathcal{U}_j^{\rightleftharpoons k}$ determined as in (2) using the mean and confidence vector $\boldsymbol{\tau}_j^{\rightleftharpoons k}$ and $\boldsymbol{\lambda}_j^{\rightleftharpoons k} = \sqrt{\boldsymbol{s}_j^{\rightleftharpoons k}}$ provided by (3.6) and (3.7).

Theorem 10. Let M, κ, and $n_j^{\rightarrow}$ be as in Theorem 8. For any $j \in \{1, 2, \ldots, k\}$, with probability at least $1 - \delta$ we have that

$$R(\mathcal{U}_j^{\rightleftharpoons k}) \leq R_j^{\infty} + \frac{M(\kappa+1)\sqrt{2\log(2m/\delta)}}{\sqrt{n_j^{\rightleftharpoons k}}} \left\|\boldsymbol{\mu}_j^{\infty}\right\|_1 \tag{3.8}$$

with $n_k^{\rightleftharpoons k} = n_k^{\rightarrow}$ and

$$n_j^{\rightleftharpoons k} \geq n_j^{\rightarrow} + n_{j+1}^{\leftarrow k} \frac{\|\boldsymbol{\sigma}_j^2\|_{\infty}}{\|\boldsymbol{\sigma}_j^2\|_{\infty} + \ n_{j+1}^{\leftarrow k}\|\boldsymbol{d}_{j+1}^2\|_{\infty}}$$

for $j \leq k-1$, where the backward ESSs satisfy $n_k^{\leftarrow k} = n_k$ and

$$n_j^{\leftarrow k} \geq n_j + n_{j+1}^{\leftarrow k} \frac{\|\boldsymbol{\sigma}_j^2\|_{\infty}}{\|\boldsymbol{\sigma}_j^2\|_{\infty} + n_{j+1}^{\leftarrow k}\|\boldsymbol{d}_{j+1}^2\|_{\infty}}.$$

Proof. See Appendix C.1. □

To the best of our knowledge, Theorem 10 provides the first performance guarantees for CL that show positive backward transfer. In particular, the bounds for forward and backward learning provided by inequality (3.8) are significantly lower than those for forward learning in Theorem 8. The ESS of each task is obtained by adding a fraction of the ESS for the succeeding task to the ESS of the corresponding task using forward learning. In particular, if $\boldsymbol{d}_j^2$ is large, the ESS is given by that with forward learning, while if $\boldsymbol{d}_j^2$ is small, the ESS is given by the sum of the ESS using forward learning and the ESS of the succeeding task.

Theorem 10 shows the increase in ESS in terms of the ESS with forward learning and the ESS of the succeeding task. The following result allows to directly quantify the ESS in terms of the sample size and the expected quadratic change.

Theorem 11. Let d, $\boldsymbol{\sigma}_j$ and n be such that $d^2 \geq \|\boldsymbol{d}_j^2\|_{\infty}$, $\|\boldsymbol{\sigma}_j^2\|_{\infty} \leq 1$, and $n \leq n_j$ for $j = 1, 2, \ldots, k$. For any $j \in \{1, 2, \ldots, k\}$, we have that the ESS in (3.8) can be taken so that it satisfies

$$n_j^{\rightleftharpoons k} \geq n\Big(1 + \frac{(1+\alpha)^{2j-1} - 1 - \alpha}{\alpha(1+\alpha)^{2j-1} + \alpha} + \frac{(1+\alpha)^{2(k-j)+1} - 1 - \alpha}{\alpha(1+\alpha)^{2(k-j)+1} + \alpha}\Big) \tag{3.9}$$

with $\alpha = \frac{2}{\sqrt{1 + \frac{4}{nd^2}} - 1}$. In particular, for $j \geq 2$, we have that

$$\begin{aligned}
n_j^{\rightleftharpoons k} &\geq n_j^{\rightarrow} + n\frac{j(k-j)}{j + 2(k-j)} \geq n\Big(1 + \frac{j-1}{3} + \frac{j(k-j)}{j+2(k-j)}\Big) && \text{if} \quad nd^2 < \frac{1}{j^2} \\
n_j^{\rightleftharpoons k} &\geq n_j^{\rightarrow} + \frac{1}{5}\sqrt{\frac{n}{d^2}} \geq n\left(1 + \frac{2}{5\sqrt{nd^2}}\right) && \text{if } \frac{1}{j^2} \leq nd^2 < 1 \\
n_j^{\rightleftharpoons k} &\geq n_j^{\rightarrow} + \frac{1}{3d^2} \geq n\left(1 + \frac{2}{3nd^2}\right) && \text{if} \quad nd^2 \geq 1.
\end{aligned}$$

Table 3.1: Datasets characteristics.

Dataset	Classes	Samples	Tasks
Yearbook	2	37,921	126
ImageNet Noise	2	12,000	10
DomainNet	4	6,256	6
UTKFace	2	23,500	94
Rotated MNIST	2	70,000	60
CLEAR	3	10,490	10

Proof. See Appendix C.2. □

The above theorem characterizes the increase in ESS provided by forward and backward learning in terms of the tasks' expected quadratic change.

Such increase grows monotonically with the number of preceding tasks j and with the number of succeeding tasks $k-j$ as shown in (3.9). In addition, it becomes proportional to the total number of tasks k when the expected quadratic change is smaller than $1/(j^2 n)$ and $j \geq k/2$. Figure 3.3 further illustrates the increase in ESS with respect to the sample size ($n_j^{\rightleftharpoons k}/n$) due to forward and backward learning in comparison with forward learning. Such figure displays the three intervals that are discussed in Theorems 9 and 11. In particular, the ESS significantly increases when nd^2 decreases between 1 and $1/j^2$. Note also that in most of the situations, the benefits of backward learning are achieved using only $b = k - j = 3$ backward steps.

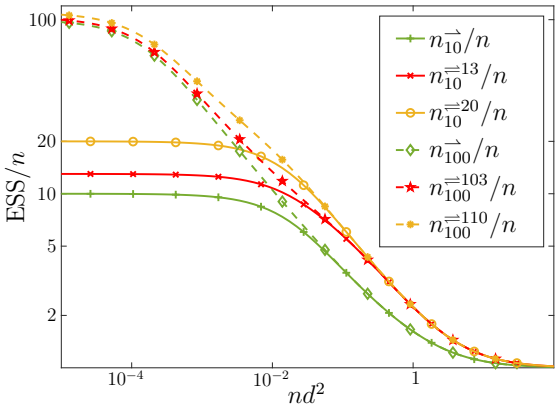


Figure 3.3: ESS increase provided by forward and backward learning.

3.5 Numerical results

In the first set of numerical results, we compare the classification performance of CL-MRCs with existing techniques using multiple datasets; in the second set of numerical results, we show the performance improvement of the presented CL-MRCs due forward and backward learning; and in the third set of additional results, we show the classification error and the running time of CL-MRCs for different hyper-parameter values.

The proposed method is evaluated using 6 public datasets: "Yearbook" [19], "ImageNet noise" [97], "UTKFaces" [99], "Rotated MNIST" [121], "DomainNet" [98], and

"CLEAR" [100]. These datasets are composed by time-dependent tasks (images with characteristics/quality/realism that change over time). The last two datasets are multi-class problems and the rest are binary. The summary of used datasets is provided in Table 3.1 that shows the number of classes, the number of samples, and the number of tasks. In the following, we further describe the tasks and the time-dependency of each dataset used.

- The "Yearbook" dataset contains portraits' photographs over time and the goal is to predict males and females. Each task corresponds to portraits from one year from 1905 to 2013.

- The "ImageNet noise" dataset contains images with increasing noise over tasks and the goal is to predict if an image is a bird or a snake. The sequence of tasks corresponds to the noise factors $[0.0, 0.4, 0.8, 1.2, 1.6, 2.0, 2.4, 2.8, 3.2, 3.6]$ [97].

- The "DomainNet" dataset contains six different domains with decreasing realism and the goal is to predict if an image is an airplane, bus, ambulance, or police car. The sequence of tasks corresponds to the six domains: real, painting, infograph, clipart, sketch, and quickdraw.

- The "UTKFaces" dataset contains face images in the wild with increasing age and the goal is to predict males and females. The sequence of tasks corresponds to face images with different ages from 0 to 116 years.

- The "Rotated MNIST" dataset contains rotated images with increasing angles over tasks and the goal is to predict if the number in an image is greater than 5 or not. Each j-th task corresponds to a rotation angle randomly selected from $\left[\frac{180(j-1)}{k}, \frac{180j}{k}\right]$ degrees where $j \in \{1, 2, \ldots, k\}$ and k is the number of tasks.

- The "CLEAR" dataset contains images with a natural temporal evolution of visual concepts in the real world and the goal is to predict if an image is soccer, hockey, or racing. Each task corresponds to one year from 2004 to 2014.

The samples in each task are randomly splitted in 100 samples for test and the rest of the samples for training. The samples used for training in the numerical results are randomly sampled from each group of training samples in each repetition.

For all methods, instances are represented by pixel values in "Rotated MNIST" dataset, and by the last layer of the ResNet18 pre-trained network [102] in the remaining datasets. The proposed CL-MRC method is compared with 4 CL techniques: gradient episodic memory (GEM) [11], meta-experience replay (MER) [49], efficient continual learning algorithm (ELLA) [7], and elastic weight consolidation (EWC) [14]. The hyper-parameters in all methods are set to the default values provided by the authors. CL-MRCs are implemented using $b = 3$ backward steps, the confidence vector is obtained using $\lambda_0 = 1$ in (7), the expected quadratic change $\boldsymbol{d}_j^2$ is estimated using

Table 3.2: Classification error and standard deviation of the proposed CL-MRC method in comparison with the existing techniques.

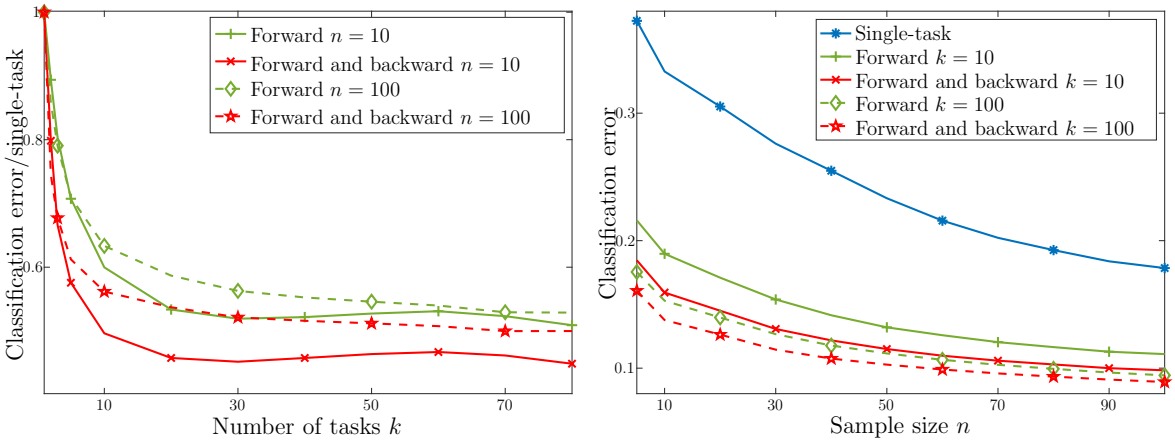
Dataset	Yearbook		ImageNet noise		DomainNet		UTKFaces		Rotated MNIST		CLEAR	
n	10	100	10	100	10	100	10	100	10	100	10	100
GEM	.18 ± .03	.17 ± .03	.39 ± .08	.13 ± .07	.69 ± .05	.53 ± .10	.12 ± .00	.12 ± .00	**.36** ± .06	.28 ± .02	.57 ± .10	.09 ± .02
MER	.16 ± .03	.10 ± .01	.17 ± .03	.10 ± .01	.38 ± .04	**.26** ± .04	.17 ± .09	.11 ± .01	.37 ± .09	.45 ± .10	.10 ± .03	**.05** ± .02
ELLA	.45 ± .09	.43 ± .10	.48 ± .05	.47 ± .04	.67 ± .05	.67 ± .05	.19 ± .12	.17 ± .11	.48 ± .05	.47 ± .05	.61 ± .06	.60 ± .05
EWC	.47 ± .05	.27 ± .06	.47 ± .04	.46 ± .06	.75 ± .04	.74 ± .05	.12 ± .00	.12 ± .00	.48 ± .01	.40 ± .01	.65 ± .03	.62 ± .04
CL-MRC	**.13** ± .04	**.08** ± .02	**.15** ± .03	**.09** ± .01	**.34** ± .06	.28 ± .01	**.10** ± .01	**.10** ± .00	**.36** ± .01	**.21** ± .00	**.09** ± .03	**.05** ± .02

$W = 2$ in (3.3), and the classifier parameters are obtained using an accelerated subgradient method based on Nesterov approach as described in Section 2.4. The subgradient method is implemented using $K = 2000$ iterations and a warm-start that initializes the classifier parameters in (2.12) with the solution obtained for the closest task. We use the same hyper-parameters for all the results in this section for fair comparison with existing methods and to show that the techniques presented do not require a careful fine-tuning. In the following, among other additional results, we study the change in classification error and processing time achieved by varying the number b of backward steps.

In the first set of numerical results, we compare the performance of the proposed CL-MRCs with the state-of-the-art techniques for $n = 10$ and $n = 100$ samples per task. These numerical results are obtained computing the average classification error over all the tasks in 50 random instantiations of data samples. As can be observed in Table 3.2, CL-MRCs can significantly improve performance in time-dependent tasks with respect to existing methods.

In the second set of numerical results, we analyze the contribution of forward and backward learning to the final performance of CL-MRCs. In particular, we show the relationship among classification error, number of tasks, and sample size for single-task, forward, and forward and backward learning. These numerical results are obtained averaging, for each number of tasks and sample size, the classification errors achieved with 10 random instantiations of data samples in "Yearbook" dataset. Figure 3.4a shows the classification error of CL-MRC method divided by the classification error of single-task learning for different number of tasks with $n = 10$ and $n = 100$ sample sizes. Such figure shows that forward and backward learning can significantly improve performance as tasks arrive. In addition, Figure 3.4b shows the classification error of CL-MRC method for different sample sizes with $k = 10$ and $k = 100$ tasks. Such figure shows that forward and backward learning for $k = 100$ tasks using $n = 10$ samples achieves significantly better results than single-task learning using $n = 100$ samples. In particular, the methods proposed can effectively exploit backward learning that results in enhanced classification error in all the experimental results.

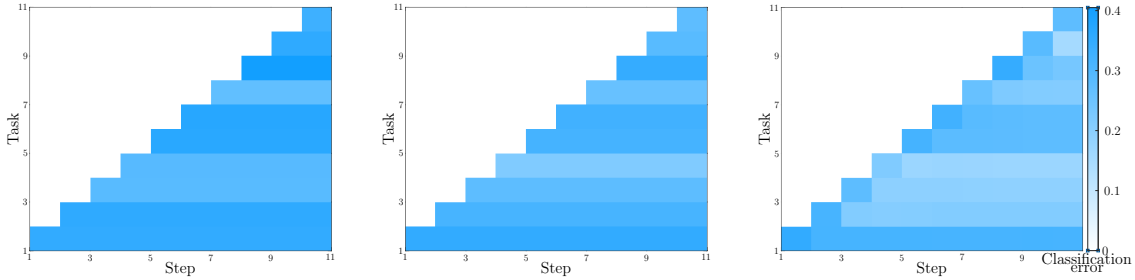
Figure 3.5 shows the classification error of CL-MRCs per step and task with single-task learning, forward learning, and forward and backward learning using the "Yearbook" dataset. Such figure shows that forward and backward learning can improve performance of preceding tasks, while forward learning and single task learning main-



(a) Classification error per number of tasks in "Yearbook" dataset.

(b) Classification error per sample size in "Yearbook" dataset.

Figure 3.4: Forward and backward learning can sharply boost performance and ESS as tasks arrive.



(a) Single task.

(b) Forward.

(c) Forward and backward.

Figure 3.5: Forward and backward learning can improve performance of preceding tasks.

tain the same performance over time.

In the third set of additional results, we further assess the change in classification error and the running time of CL-MRCs varying the hyper-parameters. Table 3.3 shows the classification error of CL-MRCs varying the values of hyper-parameter for the window size W and the number of backward steps b. As shown in the table, the proposed CL-MRCs do not require a careful fine-tuning of hyper-parameters and similar performances are obtained by using different values. In addition, Table 3.4 shows the mean running time per task in seconds of CL-MRCs for $b = 1, 2, \ldots, 5$ backward steps in comparison with the state-of-the-art techniques. Such table shows that the methods proposed for backward learning do not require a significant increase in complexity. In addition, Table 3.4 shows that the running time of the proposed method is similar to that of other state-of-the-art methods.

Table 3.3: Classification error of the proposed CL-MRC method varying W and b.

Dataset	Yearbook		ImageNet noise		DomainNet		UTKFaces		Rotated MNIST		CLEAR	
n	10	100	10	100	10	100	10	100	10	100	10	100
$W = 2$	.13	.08	.15	.09	.34	.28	.10	.10	.36	.21	.09	.05
$W = 4$	.13	.09	.15	.09	.32	.27	.10	.10	.35	.21	.09	.05
$W = 6$	.13	.09	.15	.09	.33	.28	.10	.10	.36	.21	.09	.06
$b = 1$	.14	.10	.15	.09	.36	.29	.10	.10	.36	.22	.10	.05
$b = 2$	.14	.09	.15	.09	.35	.28	.10	.10	.36	.22	.09	.05
$b = 3$	.13	.08	.15	.09	.34	.28	.10	.10	.36	.21	.09	.05
$b = 4$	.13	.08	.15	.08	.34	.28	.10	.10	.36	.21	.09	.05
$b = 5$	.13	.08	.15	.08	.34	.28	.10	.10	.36	.21	.08	.05

Table 3.4: Running time in seconds of CL-MRC method in comparison with the state-of-the-art techniques.

Dataset	Yearbook		ImageNet noise		DomainNet		UTKFaces		Rotated MNIST		CLEAR	
n	10	100	10	100	10	100	10	100	10	100	10	100
ELLA	0.066	0.070	0.073	0.073	0.054	0.065	0.059	0.062	0.176	0.198	0.077	0.073
GEM	0.098	0.476	0.010	0.037	0.005	0.020	0.056	0.310	0.180	1.094	0.009	0.034
MER	0.166	3.726	0.079	1.052	0.066	0.900	0.127	3.458	0.211	4.092	0.074	1.063
EWC	0.358	3.031	0.032	0.252	0.018	0.155	0.245	2.246	0.587	5.296	0.031	0.248
CL-MRC $b = 1$	0.094	0.324	0.259	0.490	0.518	8.463	0.108	0.348	0.135	0.471	0.235	1.310
CL-MRC $b = 2$	0.105	0.397	0.261	0.531	0.543	8.983	0.115	0.401	0.165	0.599	0.252	1.406
CL-MRC $b = 3$	0.133	0.487	0.284	0.561	0.542	9.514	0.133	0.488	0.209	0.737	0.255	1.469
CL-MRC $b = 4$	0.167	0.582	0.304	0.585	0.559	9.699	0.156	0.573	0.249	0.877	0.271	1.595
CL-MRC $b = 5$	0.184	0.664	0.438	0.601	0.571	9.921	0.190	0.664	0.288	1.010	0.360	1.693

3.6 Conclusion

This chapter proposes CL-MRCs that effectively perform forward and backward learning and account for time-dependent tasks. CL-MRCs carefully avoid the repeated usage of the same information from the ever-increasing sequence of succeeding tasks. In addition, this chapter analytically characterizes the increase in ESS achieved by the proposed forward and backward learning techniques in terms of the tasks' expected quadratic change and number of tasks. The numerical results assess the performance improvement of CL-MRC methodology with respect to existing methods using multiple datasets, sample sizes, and number of tasks. The proposed methodology for CL with time-dependent tasks can lead to techniques that further approach the humans' ability to learn from few examples and to continuously improve on tasks that arrive over time.

Chapter 4

Probabilistic Load Forecasting Based on Adaptive Online Learning

4.1 Introduction

Load forecasting is crucial for multiple energy management tasks such as scheduling generation capacity, planning supply and demand, and minimizing energy trade costs [122, 123]. The importance of load forecasting is growing significantly in recent years due to the increasing development of power systems and smart grids [123]. In addition, accurate load forecasting has a beneficial impact in environment and economy by reducing energy waste and purchase [124].

Forecasting methods are enabled by exploiting consumption patterns related to multiple factors such as past loads, hours of day, days of week, holidays, and temperatures [123–128]. Accurate forecasting is hindered by intrinsic uncertainties in load demand and dynamic changes in consumption patterns [129, 130]. These problems are becoming more relevant in recent years due to the integration of renewable energies, electric cars, and microgrids [131–135]. Uncertainties in load demand cannot be assessed by methods that obtain single-value forecasts, and dynamic changes in consumption patterns cannot be captured by methods based on offline learning of static models. On the other hand, probabilistic forecasts can evaluate load uncertainty and are essential for optimal stochastic decision making (e.g., unit commitment [136]) [137] while online learning is necessary to harness dynamic changes in consumption patterns [138]. Figure 4.1 illustrates how changes in consumption patterns affect the performance of methods based on offline and online learning. The top figure shows the two-peak consumption pattern that both methods learn on day t_0, and the middle figure shows how both methods accurately forecast until a flatter pattern emerges on day $t_0 + 3d$. Then, methods based on offline learning cannot adapt to the new pattern, while methods based on online learning correctly adapt to such change.

Most conventional techniques for load forecasting obtain single-value forecasts based on offline learning. Such techniques can be classified in three main groups: techniques

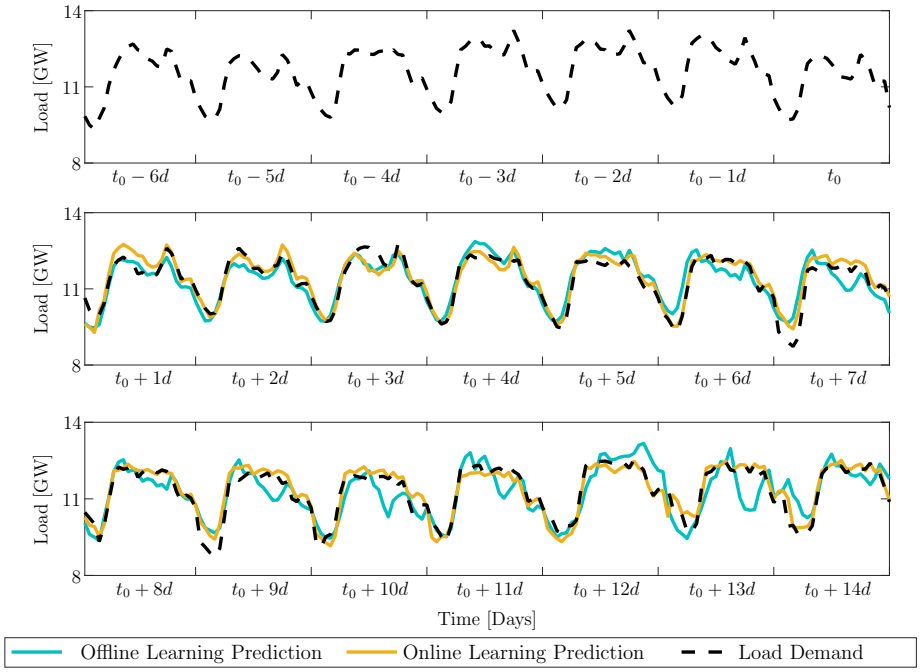


Figure 4.1: The offline learning method cannot capture the change from two-peak to flatter pattern on day t_0+3d, while the online learning method can harness such change.

based on statistical methods (e.g., linear regression (LR) [128], autoregressive moving average (ARMA) [139, 140], autoregressive integrated moving average (ARIMA) [141], and seasonal autoregressive integrated moving average (SARIMA) [142, 143]); techniques based on machine learning methods (e.g., deep learning [144], neural networks [145], and support vector machines (SVMs) [128, 146]); and techniques based on weighted combinations of several forecasts [147–149]. Existing techniques that obtain probabilistic forecasts are based on offline learning, while those based on online learning obtain single-value forecasts. Current probabilistic methods are based on Gaussian process (GP) [150] and quantile regression (QR) [151, 152]. Current online learning methods adapt to dynamic changes in consumption patterns by adjusting offline learning algorithms. In particular, such methods retrain regularly the models of conventional techniques such as ARMA [153], update the weights in combined forecasts [154], or update the smoothing functions in additive models [155].

In this chapter we present techniques for adaptive probabilistic load forecasting (APLF) that can harness changes in consumption patterns and assess load uncertainties. Specifically, the main contributions are as follows:

- We model the data using hidden Markov models (HMMs) and develop online learning techniques for APLF that update HMM parameters recursively.
- We develop sequential prediction techniques for APLF that obtain probabilistic

forecasts using the most recent HMM parameters.

- We describe in detail the efficient implementation of the steps for online learning and probabilistic prediction in APLF method.
- We quantify the performance improvement provided by the method presented in comparison with existing load forecasting techniques under multiple scenarios.

The rest of this chapter is organized as follows. Section 4.2 describes the problem of load forecasting and introduces the performance metrics. In Section 4.3, we present the theoretical results for APLF learning and prediction. Section 4.4 compares the procedures of offline learning and online learning, and describes in detail the implementation of APLF. The performance of APLF and existing techniques is compared in Section 4.5 under multiple scenarios. Finally, Section 4.6 draws the conclusions.

Notations $N(x;\mu,\sigma)$ denotes the Gaussian density function of the variable x with mean μ and standard deviation σ; $p(x|y)$ denotes the probability of variable x given variable y; $p(x,y)$ denotes the joint probability of variables x and y; $\mathbb{1}\{\cdot\}$ denotes the indicator function; bold lowercase letters represent vectors; bold capital letters represent matrices; $\mathbf{I}_K$ denotes the $K \times K$ identity matrix; $\mathbf{0}_K$ denotes the zero vector of length K; $[\,\cdot\,]$ denotes vectors; and $[\,\cdot\,]^{\mathrm{T}}$ and $\mathbb{E}\{\cdot\}$ denote the transpose and expectation of its argument.

4.2 Problem formulation

Load forecasting methods estimate future loads given past loads and factors that affect future loads such as hours of day, days of week, and weather forecasts. Forecasting techniques determine a prediction function that assigns instance vectors $\mathbf{x}$ (predictors) to target vectors $\mathbf{y}$ (responses).

Instance vectors $\mathbf{x}$ are composed by past loads and observations related to future loads (e.g., weather forecasts), and target vectors $\mathbf{y}$ are composed by future loads. We denote load by s and load forecast by $\hat{s}$, with s_t and $\hat{s}_t$ being the load and the load forecast at time t. In addition, for each time t, we denote by $\mathbf{r}_t$ the observations vector at time t that can include data such as weather forecasts w_t. Then, for a prediction horizon L (e.g., 24 hours, 30 minutes) and prediction times $t+1,\ t\ +\ 2,\ ...,\ t+L$, the instance vector is given by $\mathbf{x} = \left[s_{t-m}, ..., s_t, \mathbf{r}_{t+1}^{\mathrm{T}}, ..., \mathbf{r}_{t+L}^{\mathrm{T}}\right]^{\mathrm{T}}$, the target vector is given by $\mathbf{y} = [s_{t+1}, s_{t+2}, ..., s_{t+L}]^{\mathrm{T}}$, and the vector of load forecasts is given by $\hat{\mathbf{y}} = [\hat{s}_{t+1}, \hat{s}_{t+2}, ..., \hat{s}_{t+L}]^{\mathrm{T}}$. Furthermore, each time t is categorized by a calendar variable $c(t) \in \{1, 2, ..., C\}$ that describes time factors affecting load demand such as hour of day, day of week, month of year, and holiday. The calendar variable is used to model separately loads corresponding with each calendar type $c(t) \in \{1, 2, ..., C\}$ as described in Section 4.3. Conventional techniques such as LR [128] and SVMs [146] use instance vectors composed by past loads, observations, and calendar variables.

The proposed APLF method uses instance vectors composed by one past load and observations.

Forecasting methods determine prediction functions using training samples formed by pairs of vectors $\mathbf{x}$ and $\mathbf{y}$. Offline learning algorithms determine a static prediction function f using training samples obtained up to time t_0, $(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), ..., (\mathbf{x}_{t_0}, \mathbf{y}_{t_0})$, while online learning algorithms determine prediction functions f_t using all available training samples at $t \geq t_0$, $(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), ..., (\mathbf{x}_t, \mathbf{y}_t)$. Therefore, the static prediction function f cannot adapt to changes in consumption patterns that occur after time t_0, while prediction functions f_t can adapt to patterns' changes using the latest information (see Figure 4.1).

Performance of forecasting algorithms is evaluated in terms of accuracy using the absolute value of prediction errors:

$$e = |s - \hat{s}| \tag{4.1}$$

while probabilistic performance can be evaluated using metrics such as pinball losses [156] and calibration [157]. Overall prediction errors are commonly quantified using root mean square error (RMSE) given by

$$\text{RMSE} = \sqrt{\mathbb{E}\left\{|s - \hat{s}|^2\right\}}$$

and mean average percentage error (MAPE) given by

$$\text{MAPE} = 100 \cdot \mathbb{E}\left\{\frac{|s - \hat{s}|}{s}\right\}.$$

The pinball loss of the q-th quantile forecast $\hat{s}^{(q)}$ is given by

$$L\big(s, \hat{s}^{(q)}\big) = \begin{cases} q\left(s - \hat{s}^{(q)}\right) & \text{if} \quad s \geq \hat{s}^{(q)} \\ (1-q)\left(\hat{s}^{(q)} - s\right) & \text{if} \quad s < \hat{s}^{(q)} \end{cases}$$

and the overall pinball loss is commonly quantified by the average over all quantiles. The calibration of the q-th quantile forecast $\hat{s}^{(q)}$ is given by

$$C(q) = \mathbb{E}\left\{\mathbb{1}_{s \leq \hat{s}^{(q)}}\right\}$$

and quantifies the probability with which the load is smaller than the quantile forecast $\hat{s}^{(q)}$. Finally, the expected calibration error (ECE) is given by

$$\text{ECE} = \mathbb{E}\left\{|q - C(q)|\right\}$$

and quantifies the overall calibration error of probabilistic forecasts.

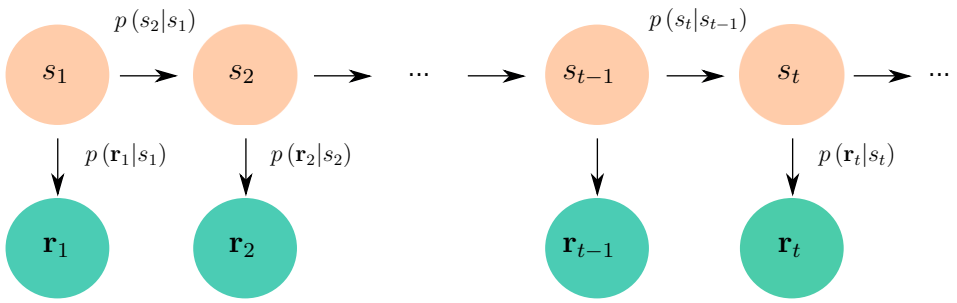


Figure 4.2: Hidden Markov model for sequences $\{s_t\}_{t\geq 1}$ and $\{\mathbf{r}_t\}_{t\geq 1}$ characterized by conditional distributions $p(s_t|s_{t-1})$ and $p(\mathbf{r}_t|s_t)$.

4.3 Models and theoretical results

This section first describes the HMM that models loads and observations, we then develop the techniques for online learning and probabilistic forecasting.

We model the relationship between the loads $\{s_t\}_{t\geq 1}$ and observations $\{\mathbf{r}_t\}_{t\geq 1}$ using HMMs also known as state-space models [158, 159]. Such models allow to predict hidden states from past states and observations, and are determined by the conditional distribution $p(s_t|s_{t-1})$ that represents the relationship between two following loads, and the conditional distribution $p(\mathbf{r}_t|s_t)$ that represents the relationship between each load and observations vector (see Figure 4.2). We model the sequence of loads and observations as a non-homogeneous HMM so that both conditional distributions change in time. Such dynamic modelling allows to adapt to changes in consumption patterns.

The conditional distributions $p(s_t|s_{t-1})$ and $p(\mathbf{r}_t|s_t)$ are modeled using Gaussian distributions with mean $\mathbf{u}^{\mathrm{T}}\boldsymbol{\eta}$ and standard deviation σ, where $\mathbf{u}$ is a known feature vector and parameters $\boldsymbol{\eta}$, σ are different for each calendar type $c(t) \in \{1, 2, ..., C\}$. For each $c = c(t)$, we denote by $\mathbf{u}_s^{\mathrm{T}}\boldsymbol{\eta}_{s,c}$ and $\sigma_{s,c}$ the mean and standard deviation that determine the conditional distribution of load at time t given load at time $t-1$ that is

$$p(s_t|s_{t-1}) = N\left(s_t; \mathbf{u}_s^{\mathrm{T}}\boldsymbol{\eta}_{s,c}, \sigma_{s,c}\right) \tag{4.2}$$

with $\boldsymbol{\eta}_{s,c} \in \mathbb{R}^2$, $\sigma_{s,c} \in \mathbb{R}$, and $\mathbf{u}_s^{\mathrm{T}} = [1, s_{t-1}]^{\mathrm{T}}$. In addition, for each $c = c(t)$, we denote by $\mathbf{u}_r^{\mathrm{T}}\boldsymbol{\eta}_{r,c}$ and $\sigma_{r,c}$ the mean and standard deviation that determine the conditional distribution of load at time t given observations at time t. Hence, assuming there is no prior knowledge available for the loads, we have that

$$p(\mathbf{r}_t|s_t) \propto p(s_t|\mathbf{r}_t) = N\left(s_t; \mathbf{u}_r^{\mathrm{T}}\boldsymbol{\eta}_{r,c}, \sigma_{r,c}\right) \tag{4.3}$$

with $\boldsymbol{\eta}_{r,c} \in \mathbb{R}^R$, $\sigma_{r,c} \in \mathbb{R}$, and $\mathbf{u}_r = u_r(\mathbf{r}_t) \in \mathbb{R}^R$. The proposed method can consider general functions $u_r(\cdot)$ and observations $\mathbf{r}_t$. In cases where the observations vector is high dimensional, APLF method can use a function $u_r(\cdot)$ that reduces the dimensionality of observations. For instance, if $\mathbf{r} \in \mathbb{R}^N$, $u_r(\mathbf{r}) \in \mathbb{R}^R$ can be the result of applying a dimensionality reduction method such as principal component analysis (PCA). In the

experimental results of Section 4.5, we use a simple function $u_r(\cdot)$ that returns binary vectors and encodes weather shifts.

Using the above models, at each time t the HMM describing the sequences of loads and observations is characterized by parameters

$$\Theta = \{\boldsymbol{\eta}_{s,c}, \sigma_{s,c}, \boldsymbol{\eta}_{r,c}, \sigma_{r,c} : c = 1, 2, ..., C\} \tag{4.4}$$

where $\boldsymbol{\eta}_{s,c}, \sigma_{s,c}$ characterize the conditional distribution $p(s_t|s_{t-1})$ and $\boldsymbol{\eta}_{r,c}, \sigma_{r,c}$ characterize the conditional distribution $p(\mathbf{r}_t|s_t)$ for times t with calendar type $c = c(t)$.

The parameters for each calendar type and conditional distribution can be obtained by maximizing the weighted log-likelihood of all the loads obtained at times with the same calendar type. Specifically, if $s_{t_1}, s_{t_2}, ..., s_{t_n}$ are loads obtained at times with calendar type $c \in \{1, 2, ..., C\}$, i.e., $c = c(t_1) = c(t_2) = ... = c(t_n)$, and $\mathbf{u}_{t_1}, \mathbf{u}_{t_2}, ..., \mathbf{u}_{t_n}$ are the corresponding feature vectors for parameters $\boldsymbol{\eta}_{s,c}, \sigma_{s,c}$ as given in (4.2) or for parameters $\boldsymbol{\eta}_{r,c}, \sigma_{r,c}$ as given in (4.3), the exponentially weighted log-likelihood of loads up to time t_i, for $i = 1, 2, ..., n$, is given by

$$L_i(\boldsymbol{\eta}, \sigma) = \sum_{j=1}^{i} \lambda^{i-j} \log N(s_{t_j}; \mathbf{u}_{t_j}^{\mathrm{T}} \boldsymbol{\eta}, \sigma) \tag{4.5}$$

where weights λ^{i-j}, $j = 1, 2, ..., i$, allow to increase the influence of the most recent data using a parameter $\lambda \in (0, 1)$ that is commonly known as forgetting factor. The maximization of (4.5) is a convex optimization problem since $L_i(\boldsymbol{\eta}, \sigma)$ is a concave function because Gaussian distributions are log-concave and $\lambda^{i-j} > 0$ for any i, j. In addition, the maximum of (4.5) is unique as long as its Hessian is negative definite which happens for any $i \geq i_0$ such that

$$\mathbf{H}_{i_0} = \sum_{j=1}^{i_0} \lambda^{i_0-j} \mathbf{u}_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \tag{4.6}$$

is a non-singular matrix.

The next Theorem shows that the maximization of the weighted log-likelihood in (4.5) can be solved recursively using parameters given by

$$\boldsymbol{\eta}_i = \boldsymbol{\eta}_{i-1} + \frac{\mathbf{P}_{i-1}\mathbf{u}_{t_i}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}} \left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right) \tag{4.7}$$

$$\sigma_i = \sqrt{\sigma_{i-1}^2 - \frac{1}{\gamma_i}\left(\sigma_{i-1}^2 - \frac{\lambda\left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right)^2}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\right)} \tag{4.8}$$

with

$$\mathbf{P}_i = \frac{1}{\lambda}\left(\mathbf{P}_{i-1} - \frac{\mathbf{P}_{i-1}\mathbf{u}_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\right) \tag{4.9}$$

$$\gamma_i = 1 + \lambda\gamma_{i-1}. \tag{4.10}$$

Theorem 12. Let i_0 be an integer such that the matrix $\mathbf{H}_{i_0}$ given by (4.6) is non-singular, and $\boldsymbol{\eta}_i^* \in \mathbb{R}^K$, $\sigma_i^* \in \mathbb{R}$ be parameters that maximize the weighted log-likelihood given by (4.5) with $N(s_{t_j}; \mathbf{u}_{t_j}^{\mathrm{T}} \boldsymbol{\eta}_i^*, \sigma_i^*) \leq M$ for any $j \leq i \leq n$.

If parameters $\boldsymbol{\eta}_i \in \mathbb{R}^K$, $\sigma_i \in \mathbb{R}$ are given by the recursions in (4.7)-(4.10) for $i > 0$ with $\boldsymbol{\eta}_0 = \mathbf{0}_K$, any σ_0, $\mathbf{P}_0 = \mathbf{I}_K$, and $\gamma_0 = 0$. Then, we have that

$$L_i\left(\boldsymbol{\eta}_i^*, \sigma_i^*\right) - L_i\left(\boldsymbol{\eta}_i, \sigma_i\right) \leq \pi M^2 \left\|\boldsymbol{\eta}_i^*\right\|^2 \lambda^i \tag{4.11}$$

for any $i \geq i_0$.

In addition, if parameters $\boldsymbol{\eta}_i \in \mathbb{R}^K$, $\sigma_i \in \mathbb{R}$ are given by the recursions in (4.7)-(4.10) for $i > i_0$ with

$$\boldsymbol{\eta}_{i_0} = \mathbf{P}_{i_0} \left(\sum_{j=1}^{i_0} \lambda^{i_0 - j} s_{t_j} \mathbf{u}_{t_j} \right) \tag{4.12}$$

$$\sigma_{i_0} = \sqrt{\frac{1}{\gamma_{i_0}} \left(\sum_{j=1}^{i_0} \lambda^{i_0 - j} s_{t_j}^2 - \sum_{j=1}^{i_0} \lambda^{i_0 - j} s_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \boldsymbol{\eta}_{i_0} \right)} \tag{4.13}$$

$$\mathbf{P}_{i_0} = \left(\mathbf{H}_{i_0}\right)^{-1} \tag{4.14}$$

$$\gamma_{i_0} = \sum_{j=1}^{i_0} \lambda^{i_0 - j}. \tag{4.15}$$

Then, we have that $\boldsymbol{\eta}_i = \boldsymbol{\eta}_i^*$ and $\sigma_i = \sigma_i^*$ for any $i \geq i_0$.

Proof. See appendix D.1. □

The first part of the above result shows that parameters given by the recursions (4.7)-(4.10) for $i > 0$ with $\boldsymbol{\eta}_0 = \mathbf{0}_K$, any σ_0, $\mathbf{P}_0 = \mathbf{I}_K$, and $\gamma_0 = 0$, approximately maximize the weighted log-likelihood. In addition, the log-likelihood difference with respect to the maximum given by (4.11) decreases exponentially fast with the number of iterations i since $\lambda \in (0, 1)$. The second part shows that parameters given by the recursions (4.7)-(4.10) for $i \geq i_0$ with $\boldsymbol{\eta}_{i_0}, \sigma_{i_0}$ given by (4.12)-(4.15), maximize the weighted log-likelihood for any $i \geq i_0$. Note that the parameters are updated in recursions (4.7)-(4.10) by adding a correction to the previous parameters $\boldsymbol{\eta}_{i-1}$ and σ_{i-1}. Such correction is proportional to the fitting error of the previous parameter $s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}} \boldsymbol{\eta}_{i-1}$ so that parameters are updated depending how well they fit the most recent data.

Recursion (4.7) for parameters describing means is similar to that used for the recursive minimization of weighted least squares [160]. The main technical novelty in Theorem 12 lies in recursion (4.8) for parameters describing standard deviations, and inequality (4.11) describing the near-optimality of parameters initialized with $\boldsymbol{\eta}_0 = \mathbf{0}_K$, any σ_0, $\mathbf{P}_0 = \mathbf{I}_K$, and $\gamma_0 = 0$. Existing techniques for least squares only allow to recursively obtain parameters describing means, while Theorem 12 allows to recursively obtain parameters describing both means and standard deviations. Such generalization is of practical relevance since it allows to obtain probabilistic models determined by

time-changing means and standard deviations. In addition, existing techniques address the possible singularity of matrix (4.6) during the initial steps by adding a regularization term in (4.5), but such approach cannot be used to obtain standard deviations. The bound in (4.11) guarantees that parameters given by recursions (4.7)-(4.10) are close to be optimal when initialized with $\boldsymbol{\eta}_0 = \mathbf{0}_K$, any σ_0, $\mathbf{P}_0 = \mathbf{I}_K$ and $\gamma_0 = 0$, and are optimal when initialized as given by (4.12)-(4.15).

The above Theorem enables the adaptive online learning of parameters Θ described in (4.4). As detailed in Section 4.4, Theorem 12 allows to update parameters $\boldsymbol{\eta}_{s,c}$, $\sigma_{s,c}$ and $\boldsymbol{\eta}_{r,c}$, $\sigma_{r,c}$ using the recursions (4.7) and (4.8) every time new loads and observations are obtained corresponding with calendar type c. Such parameters are updated using their previous values and the states variables given by (4.9) and (4.10). In the following, we denote $\mathbf{P}_{s,c}$ and $\gamma_{s,c}$ (resp. $\mathbf{P}_{r,c}$ and $\gamma_{r,c}$) the state variables required to update parameters $\boldsymbol{\eta}_{s,c}$ and $\sigma_{s,c}$ (resp. $\boldsymbol{\eta}_{r,c}$ and $\sigma_{r,c}$) for $c = 1, 2, ..., C$, and we denote by Γ the list composed by those state variables, that is

$$\Gamma = \{\mathbf{P}_{s,c}, \gamma_{s,c}, \mathbf{P}_{r,c}, \gamma_{r,c} : c = 1, 2, ..., C\}. \tag{4.16}$$

In addition, we denote λ_s (resp. λ_r) the forgetting factor required to update parameters $\boldsymbol{\eta}_{s,c}$ and $\sigma_{s,c}$ (resp. $\boldsymbol{\eta}_{r,c}$ and $\sigma_{r,c}$), for $c = 1, 2, ..., C$.

The previous result describes how to update HMM parameters using the most recent data, the next result shows how to obtain probabilistic forecasts using the HMM characterized by parameters Θ.

Theorem 13. If $\{s_t, \mathbf{r}_t\}_{t\geq 1}$ is an HMM characterized by parameters Θ as described in (4.4). Then, for $i = 1, 2, ..., L$

$$p\left(s_{t+i} | s_t, \mathbf{r}_{t+1}, ..., \mathbf{r}_{t+i}\right) = N\left(s_{t+i}; \hat{s}_{t+i}, \hat{e}_{t+i}\right) \tag{4.17}$$

where means $\hat{s}_{t+1}, \hat{s}_{t+2}, ..., \hat{s}_{t+L}$ and standard deviations $\hat{e}_{t+1}, \hat{e}_{t+2}, ..., \hat{e}_{t+L}$ can be computed by the following recursions

$$\begin{aligned} c &= c\left(t+i\right), \hat{\mathbf{u}}_s = \left[1, \hat{s}_{t+i-1}\right]^{\mathrm{T}}, \mathbf{u}_r = u_r\left(\mathbf{r}_{t+i}\right) \\ \hat{s}_{t+i} &= \frac{\hat{\mathbf{u}}_s^{\mathrm{T}} \boldsymbol{\eta}_{s,c} \sigma_{r,c}^2 + \mathbf{u}_r^{\mathrm{T}} \boldsymbol{\eta}_{r,c} \left(\sigma_{s,c}^2 + \left(\mathbf{v}^{\mathrm{T}} \boldsymbol{\eta}_{s,c}\right)^2 \hat{e}_{t+i-1}^2\right)}{\sigma_{r,c}^2 + \sigma_{s,c}^2 + \left(\mathbf{v}^{\mathrm{T}} \boldsymbol{\eta}_{s,c}\right)^2 \hat{e}_{t+i-1}^2} \end{aligned} \tag{4.18}$$

$$\hat{e}_{t+i} = \sqrt{\frac{\sigma_{r,c}^2 \left(\sigma_{s,c}^2 + \left(\mathbf{v}^{\mathrm{T}} \boldsymbol{\eta}_{s,c}\right)^2 \hat{e}_{t+i-1}^2\right)}{\sigma_{r,c}^2 + \sigma_{s,c}^2 + \left(\mathbf{v}^{\mathrm{T}} \boldsymbol{\eta}_{s,c}\right)^2 \hat{e}_{t+i-1}^2}} \tag{4.19}$$

for $\mathbf{v} = [0, 1]^{\mathrm{T}}$, $\hat{s}_t = s_t$, $\hat{e}_t = 0$, and $i = 1, 2, ..., L$.

Proof. See appendix D.2. □

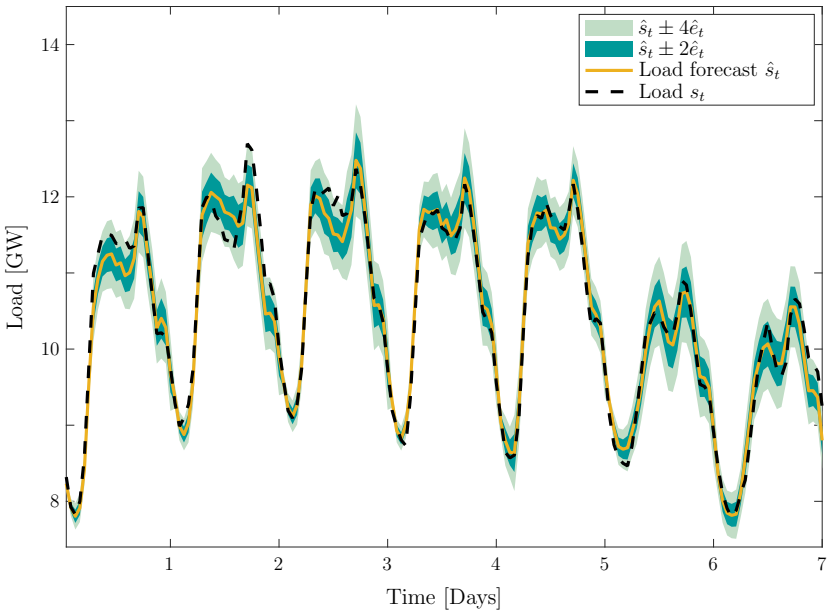


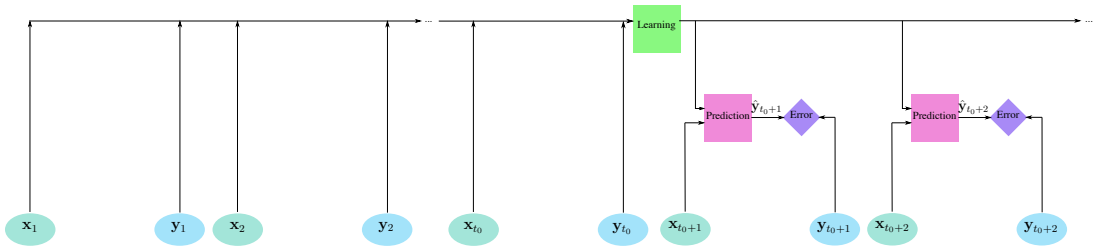
Figure 4.3: APLF method obtains load forecasts together with reliable uncertainty assessments of load demand.

The above Theorem enables to recursively obtain probabilistic load forecasts $N\left(s_{t+i}; \hat{s}_{t+i}, \hat{e}_{t+i}\right)$, for $i = 1, 2, ..., L$ that allow to quantify the probability of forecast intervals (see Figure 4.3). As detailed in Section 4.4, Theorem 13 allows to obtain load forecasts $\hat{s}_{t+i}$ together with estimates of their accuracy $\hat{e}_{t+i}$ for each $i = 1, 2, ..., L$. Such forecasts are obtained using the recursions (4.18) and (4.19) with the most recent parameters every time new instance vectors $\mathbf{x}$ are obtained. Specifically, for each $i = 1, 2, ..., L$, the probabilistic forecast at time $t + i$, $N\left(s_{t+i}; \hat{s}_{t+i}, \hat{e}_{t+i}\right)$, is obtained using 1) the probabilistic forecast $N\left(s_{t+i-1}; \hat{s}_{t+i-1}, \hat{e}_{t+i-1}\right)$ at previous time $t + i - 1$; 2) observations vector $\mathbf{r}_{t+i}$ at time $t+i$; and 3) parameters $\boldsymbol{\eta}_{s,c}, \sigma_{s,c}$ and $\boldsymbol{\eta}_{r,c}, \sigma_{r,c}$ corresponding with calendar type at time $t + i$, $c = c(t + i)$.

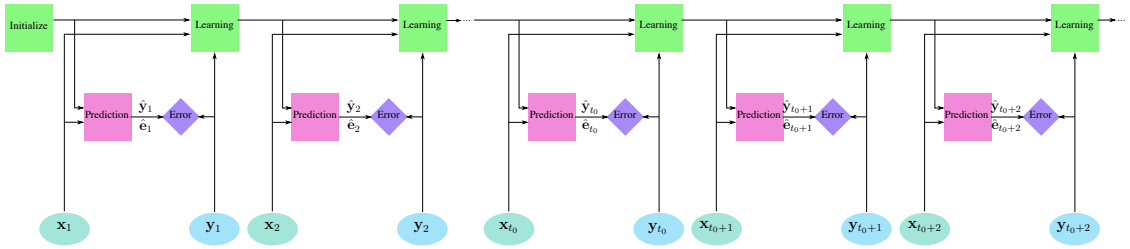
The results in this section provide theoretical guarantees for the learning and prediction steps of APLF method. The next section describes APLF in comparison with approaches based on offline learning, and details the implementation steps for learning and prediction using APLF.

4.4 Implementation

Offline learning methods obtain one model, while online learning methods obtain a sequence of models. In particular, APLF method learns a model every time a new sample is obtained. Figure 4.4 describes the block diagrams for load forecasting based on offline learning and based on APLF method.



(a) Offline learning algorithms use t_0 training samples to train the model that is later used to obtain all forecasts.



(b) APLF method updates the model every time a new sample is obtained. Then, the model used for forecasting adapts to the most recent data.

Figure 4.4: Block diagrams for offline learning algorithms and APLF method.

Load forecasting methods based on offline learning train a model using a set of samples. Later, such methods predict the load for each new instance vector using the learned model. At learning, t_0 training samples are used to obtain the model following different approaches. For instance, ARMA algorithm calculates parameters for the autoregressive, moving average, and error terms [139], while techniques based on machine learning calculate parameters determining a regression function [128, 146, 161]. At prediction, the learned model and the instance vector $\mathbf{x}_t$ are used to obtain load forecasts $\hat{\mathbf{y}}_t$ at time t, for $t > t_0$. The model used in the prediction step is the same for all times t, and the actual loads $\mathbf{y}_t$ for $t > t_0$ are only used to quantify the prediction error (see Figure 4.4a).

Load forecasting methods based on online learning train models regularly using the most recent samples. Later, such methods predict for each new instance vector using the latest learned model. At learning, training samples and possibly the previous model are used to obtain a new model following different approaches. For instance, the method proposed in [153] recalculates parameters of the ARMA algorithm, the method proposed in [154] recalculates the weights in combined forecasts, and the method proposed in [155] updates the smoothing functions in additive models. At prediction, the latest learned model and the instance vector $\mathbf{x}_t$ are used to obtain load forecasts $\hat{\mathbf{y}}_t$ at time t.

APLF is a forecasting method based on online learning that updates model parameters using recursions in Theorem 12 and obtain probabilistic forecasts as given by Theorem 13. At learning, APLF obtains the new model using the instance vec-

tor $\mathbf{x}_t$, the actual loads $\mathbf{y}_t$, and the previous model. At prediction, APLF uses the latest model to obtain load forecasts $\hat{\mathbf{y}}_t = [\hat{s}_{t+1}, \hat{s}_{t+2}, ..., \hat{s}_{t+L}]^{\mathrm{T}}$ and estimated errors $\hat{\mathbf{e}}_t = [\hat{e}_{t+1}, \hat{e}_{t+2}, ..., \hat{e}_{t+L}]^{\mathrm{T}}$ that determine probabilistic forecasts as given in (4.17). The model used in the prediction step adapts at each time t to the most recent data, and the actual loads $\mathbf{y}_t$ are not only used to quantify the error, but also to update the model (see Figure 4.4b).

Algorithm 10 and Algorithm 11 detail the efficient implementation of the learning and prediction steps of APLF. The corresponding source code in Python and Matlab languages is publicly available on the web `https://github.com/MachineLearningBCAM/Load-forecasting-IEEE-TPWRS-2020`. The running times of Algorithm 10 and Algorithm 11 are amenable for real-time implementation with very low latency since APLF has memory complexity $O(CR^2)$, the learning step has computational complexity $O(LR^3)$, and the prediction step has computational complexity $O(LR)$. Note that the values of R are small in practice, for instance we use $R = 3$, $L = 24$, and $C = 48$ in the numerical results of Section 4.5. Algorithm 10 follows recursions given in Theorem 12 for parameters Θ and state variables Γ described in (4.4) and (4.16), respectively. Specifically, such algorithm updates parameters $\boldsymbol{\eta}_{s,c}, \sigma_{s,c}$ and $\boldsymbol{\eta}_{r,c}, \sigma_{r,c}$ as well as state variables $\mathbf{P}_{s,c}, \gamma_{s,c}$ and $\mathbf{P}_{r,c}, \gamma_{r,c}$ using instances and actual loads with calendar type c. Algorithm 11 follows recursions given in Theorem 13 using the parameters Θ described in (4.4) and the new instance vector. Specifically, such algorithm obtains L load forecasts and L estimates of their accuracy using the latest parameters $\boldsymbol{\eta}_{s,c}, \sigma_{s,c}$ and $\boldsymbol{\eta}_{r,c}, \sigma_{r,c}$ and the corresponding instance vector. Note that the proposed method can predict at any time of the day and use general prediction horizons L. In addition, these prediction times and horizons can change from one day to another just by modifying the corresponding inputs in Algorithm 11.

Algorithm 10 Learning step for APLF

Input: $\Theta, \Gamma, \lambda_s, \lambda_r, \mathbf{x}_t = \left[s_t, \mathbf{r}_{t+1}^{\mathrm{T}}, \mathbf{r}_{t+2}^{\mathrm{T}}, ..., \mathbf{r}_{t+L}^{\mathrm{T}}\right]^{\mathrm{T}}, \mathbf{y}_t = [s_{t+1}, s_{t+2}, ..., s_{t+L}]^{\mathrm{T}}, t$
Output: Θ, Γ

for $i = 1, 2, ..., L$ **do**
  $c \leftarrow c\,(t+i)$, $\mathbf{u}_s \leftarrow [1, s_{t+i-1}]^{\mathrm{T}}$, $\mathbf{u}_r \leftarrow u_r\,(\mathbf{r}_{t+i})$
  for $j = s, r$ **do**
    $\mathbf{P}_{j,c} \leftarrow \frac{1}{\lambda_j}\left(\mathbf{P}_{j,c} - \frac{\mathbf{P}_{j,c}\mathbf{u}_j\mathbf{u}_j^{\mathrm{T}}\mathbf{P}_{j,c}}{\lambda_j + \mathbf{u}_j^{\mathrm{T}}\mathbf{P}_{j,c}\mathbf{u}_j}\right)$
    $\gamma_{j,c} \leftarrow 1 + \lambda_j\gamma_{j,c}$
    $\sigma_{j,c} \leftarrow \sqrt{\sigma_{j,c}^2 - \frac{1}{\gamma_{j,c}}\left(\sigma_{j,c}^2 - \frac{\lambda_j\left(s_{t+i} - \mathbf{u}_j^{\mathrm{T}}\boldsymbol{\eta}_{j,c}\right)^2}{\lambda_j + \mathbf{u}_j^{\mathrm{T}}\mathbf{P}_{j,c}\mathbf{u}_j}\right)}$
    $\boldsymbol{\eta}_{j,c} \leftarrow \boldsymbol{\eta}_{j,c} + \frac{\mathbf{P}_{j,c}\mathbf{u}_j}{\lambda_j + \mathbf{u}_j^{\mathrm{T}}\mathbf{P}_{j,c}\mathbf{u}_j}\left(s_{t+i} - \mathbf{u}_j^{\mathrm{T}}\boldsymbol{\eta}_{j,c}\right)$

Algorithm 11 Prediction step for APLF

Input: $\Theta, \mathbf{x}_t = \left[s_t, \mathbf{r}_{t+1}^{\mathrm{T}}, \mathbf{r}_{t+2}^{\mathrm{T}}, ..., \mathbf{r}_{t+L}^{\mathrm{T}}\right]^{\mathrm{T}}, t$
Output: $\hat{\mathbf{y}}_t = \left[\hat{s}_{t+1}, \hat{s}_{t+2}, ..., \hat{s}_{t+L}\right]^{\mathrm{T}}, \hat{\mathbf{e}}_t = \left[\hat{e}_{t+1}, \hat{e}_{t+2}, ..., \hat{e}_{t+L}\right]^{\mathrm{T}}, N(s_{t+i}; \hat{s}_{t+i}, \hat{e}_{t+i}),\ i = 1, 2, ..., L$
$\hat{s}_t \leftarrow s_t$
$\hat{e}_t = 0$
for $i = 1, 2, ..., L$ **do**
  $c \leftarrow c\,(t+i)$, $\hat{\mathbf{u}}_s \leftarrow \left[1, \hat{s}_{t+i-1}\right]^{\mathrm{T}}$, $\mathbf{u}_r \leftarrow u_r\,(\mathbf{r}_{t+i})$
  Obtain load forecast $\hat{s}_{t+i}$ using equation (4.18)
  Obtain prediction error $\hat{e}_{t+i}$ using equation (4.19)

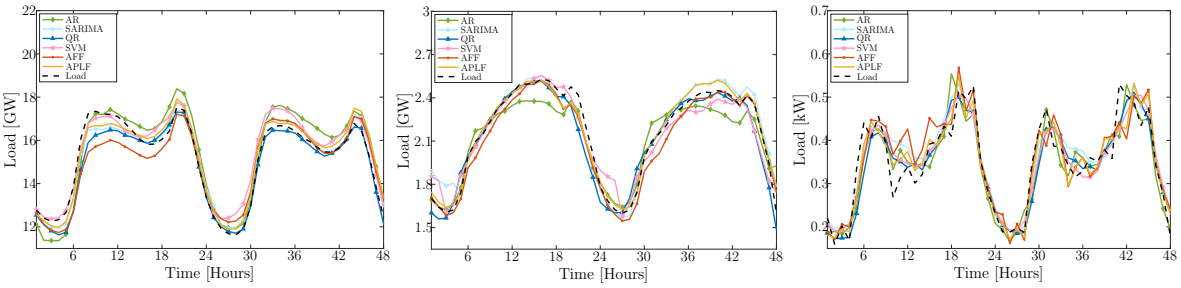
Table 4.1: RMSE and MAPE of prediction errors for APLF and 11 state-of-the-art techniques on 7 datasets.

	Large-size region				Medium-size region						Small-size region			
Method	Belgium		New Engld.		GEFCom12		GEFCom2014		Dayton		400 builds.		100 builds.	
	[GW]	[%]	[GW]	[%]	[MW]	[%]	[MW]	[%]	[GW]	[%]	[kW]	[%]	[kW]	[%]
LR	1.47	11.8	1.73	8.0	5.48	20.6	0.30	15.0	0.46	15.5	0.06	9.1	0.07	14.3
SARIMA	0.81	5.5	1.22	5.4	3.42	11.6	0.25	12.0	0.22	7.7	0.05	9.9	0.07	16.5
QR	1.05	9.2	1.17	5.6	5.46	25.5	0.29	14.7	0.20	7.1	0.08	17.6	0.12	26.8
GP	**0.52**	**4.0**	0.89	4.2	2.50	**8.5**	**0.24**	**10.6**	0.19	6.3	**0.04**	**6.8**	**0.05**	**11.2**
SVM	0.69	4.7	1.11	5.5	3.28	12.2	**0.24**	12.3	**0.17**	**5.7**	**0.04**	8.0	**0.05**	11.4
DRN	1.74	13.0	**0.52**	**2.3**	**2.17**	7.6	0.27	19.5	0.31	11.3	**0.04**	7.0	0.07	14.7
AR	0.66	5.1	1.28	5.6	3.94	16.2	0.30	18.5	0.38	13.6	**0.04**	8.9	0.07	16.9
ARNFS	1.08	9.4	1.95	10.9	4.41	17.4	0.33	18.4	0.31	14.1	**0.04**	8.1	**0.05**	11.4
ARRFFS	1.18	10.3	2.05	11.2	4.54	17.2	0.34	17.7	0.28	10.3	0.08	17.5	0.10	23.4
SFDA	1.14	8.9	1.41	10.2	5.04	16.8	0.35	14.5	0.39	21.8	0.06	13.0	0.08	18.6
AFF	0.95	6.7	1.23	5.8	2.91	10.7	0.25	15.2	0.26	9.6	0.05	10.2	0.07	14.6
APLF	**0.33**	**2.3**	**0.86**	**3.9**	**2.15**	**8.1**	**0.20**	**9.6**	**0.16**	**5.5**	**0.03**	**6.3**	**0.05**	**11.0**

4.5 Numerical results

This section first describes the datasets used for the experimentation, and then compares the performance of APLF method with respect to that of existing techniques. The first set of numerical results quantifies the prediction errors, while the second set of numerical results evaluates the performance of probabilistic load forecasts and analyzes the relationship between training size and prediction error.

Seven publicly available datasets are selected for numerical experimentation. The datasets correspond with regions that have different sizes and display different consumption patterns that change over time. Such changes can be observed in Figure 4.5 that shows load demand per hour of day and day of year during 2004 and 2005 in Dayton (US). This figure shows that consumption patterns change significantly not only for different seasons but also between consecutive weeks and between consecutive years. Therefore, methods based on static models often obtain inferior accuracies since they



(a) New England. (b) Dayton. (c) 100 buildings.

Figure 4.6: Load demand presents different consumption patterns and variability in the three regions with different sizes.

cannot adapt to dynamic changes in consumption patterns.

We group the seven datasets by size of the region: large, medium, and small. Two datasets belong to large-size regions: load demand in Belgium from 2017-2019 made available by Elia group, and load demand in New England from 2003-2014 made available by ISO-NE organization. Three datasets belong to medium-size regions: Global Energy Forecasting Competition 2012 (GEFCom2012) dataset from 2004-2007 [162], Global Energy Forecasting Competition 2014 (GEFCom2014) dataset from 2005-2011 [156], and load demand in Dayton from 2004-2016 made available by PJM interconnection. Finally, two datasets belong to small-size regions that correspond with load demand for 400 and 100 buildings in New South Wales from 2013 and are made available by the Australian Government as part of the project Smart Grid Smart Cities.

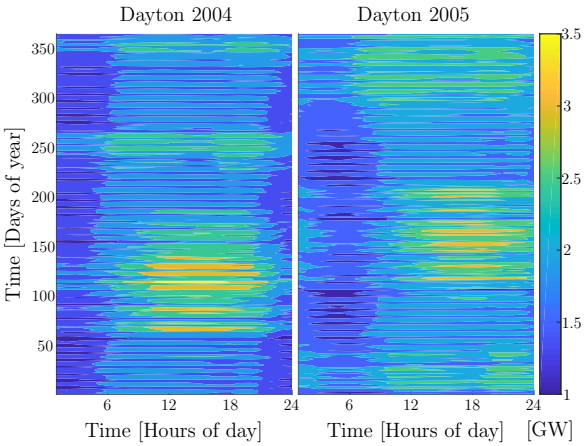
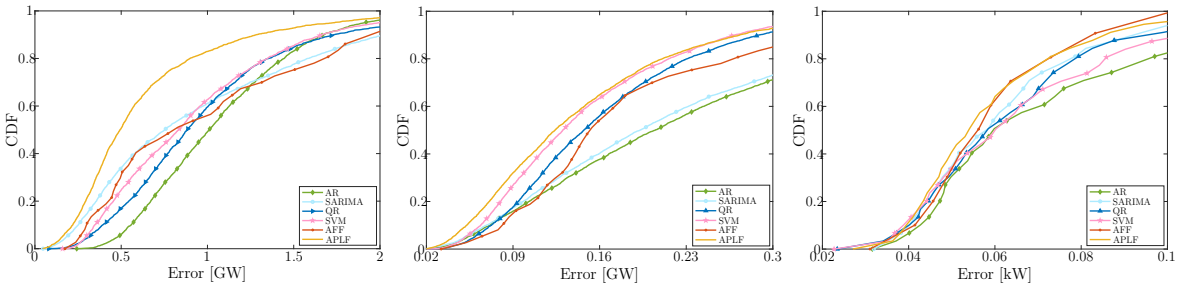


Figure 4.5: Consumption patterns can significantly change in two consecutive years.

In the numerical results, training for offline learning algorithms is done using training sizes depending on the length of datasets. Two years of data are used for training in New England and GEFCom2012 datasets; one year of data are used for training in Belgium, GEFCom2014, and Dayton datasets; and 3/4 of a year of data are used for training in 400 and 100 buildings datasets. Prediction for all algorithms is done using the rest of the data as follows. At 11 a.m. of each day, all forecasting methods obtain future loads for a prediction horizon of $L = 24$ hours hence, every vector of load forecasts is formed by forecasts obtained from 1 to L hours ahead.

APLF results are obtained using the following implementation details. The instance vector composed by past loads and observations is given by $\mathbf{x} = \left[s_t, \mathbf{r}_{t+1}^{\mathrm{T}}, ..., \mathbf{r}_{t+L}^{\mathrm{T}}\right]^{\mathrm{T}}$. The



(a) New England. (b) Dayton. (c) 100 buildings.

Figure 4.7: CDFs of prediction errors provide detailed performance comparison between APLF method and conventional techniques in three regions with different sizes.

observations vector $\mathbf{r}_t$ contains the temperature w_t at time t and the mean of past temperatures $\bar{w}_{c(t)}$ at calendar type $c(t)$, i.e., $\mathbf{r}_t = \left[w_t, \bar{w}_{c(t)}\right]^{\mathrm{T}}$. The observations vector $\mathbf{r}_t$ is represented by the feature vector $u_r(\mathbf{r}_t)$ for a function $u_r(\cdot)$ that encodes temperature shifts. Specifically, such function determines the vector $u_r(\mathbf{r}_t) = [1, \alpha_1, \alpha_2]^{\mathrm{T}}$, where α_1 (resp. α_2) takes value 1 if the temperature is above (resp. below) certain thresholds and takes value 0 otherwise, that is

$$\alpha_1, \alpha_2 = \begin{cases} 1,0 & \text{if} \quad w_t - \bar{w}_{c(t)} > W_1 \quad \text{and} \\ & \quad\; w_t > W_2 \quad \text{or} \quad w_t < W_3 \\ 0,1 & \text{if} \quad w_t - \bar{w}_{c(t)} < -W_1 \quad \text{and} \\ & \quad\; w_t > W_2 \quad \text{or} \quad w_t < W_3 \\ 0,0 & \text{otherwise} \end{cases}$$

where we take threshold values $W_1 = 20°F$, $W_2 = 80°F$, and $W_3 = 20°F$ for all datasets. The calendar information $c(t)$ specifies the type of hour: $c(t)$ from 1 to 24 indicates the hour of day of weekdays, and $c(t)$ from 25 to 48 indicates the hour of day of weekends and holidays, i.e., $c(t) \in \{1, ..., C\}$ with $C = 48$. Then, APLF method obtains parameters $\boldsymbol{\eta}_{s,c}, \sigma_{s,c}$ and $\boldsymbol{\eta}_{r,c}, \sigma_{r,c}$ for $C = 48$ calendar types as given by Algorithm 10 initialized with $\boldsymbol{\eta}_0 = \mathbf{0}_K$, any σ_0, $\mathbf{P}_0 = \mathbf{I}_K$, and $\gamma_0 = 0$. Such parameters are updated by taking forgetting factors as $\lambda_s = 0.2$ and $\lambda_r = 0.7$ for any calendar type and for all datasets.[1] Forgetting factors (λ_s, λ_r) and threshold values (W_1, W_2, W_3) are the hyper-parameters of APLF method. Hyper-parameters' values can be selected by using various methods such as cross-validation over a grid of possible values. For simplicity, we select values for hyper-parameters by inspection over one dataset and then we use the same values in all datasets. The numerical results corroborate the robustness of APLF method to the choice of hyper-parameters since we use the same hyper-parameters' values in all numerical results and with significantly different datasets.

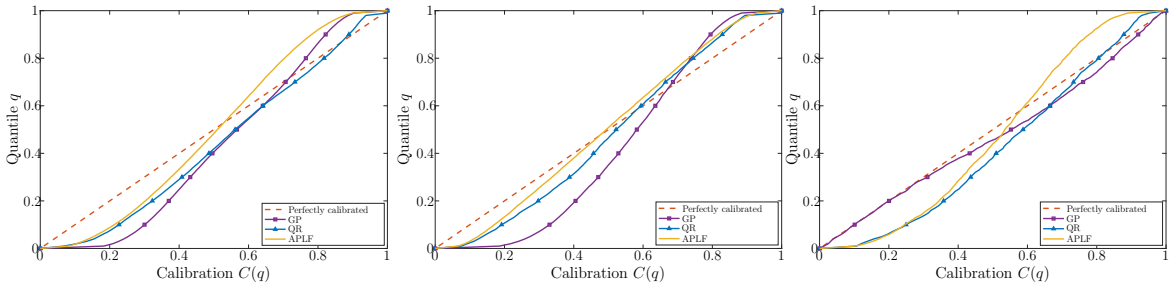
[1]Possible numerical instabilities of state matrices $\mathbf{P}$ in Algorithm 10 are addressed by their reinitialization in case their trace becomes larger than 10 similarly to methods based on recursive least squares [163].

Table 4.2: Pinball loss and ECE for APLF and 2 state-of-the-art techniques on 7 datasets.

Region	QR		GP		APLF	
	Pinball loss	ECE	Pinball loss	ECE	Pinball loss	ECE
Belgium	0.34 [GW]	0.08	0.14 [GW]	0.19	**0.11** [GW]	**0.07**
New England	0.70 [GW]	**0.07**	0.24 [GW]	0.09	**0.22** [GW]	**0.07**
GEFCom12	1.03 [MW]	**0.06**	0.78 [MW]	0.14	**0.77** [MW]	0.12
GEFCom14	0.06 [MW]	0.60	**0.05** [MW]	**0.15**	0.06 [MW]	0.19
Dayton	0.09 [GW]	0.06	**0.04** [GW]	0.12	**0.04** [GW]	**0.05**
400 buildings	0.02 [kW]	0.10	**0.01** [kW]	**0.05**	**0.01** [kW]	0.07
100 buildings	0.03 [kW]	0.08	**0.01** [kW]	**0.03**	**0.01** [kW]	0.08

APLF method is compared with 11 state-of-the-art techniques based on statistical methods, machine learning, and weighted combination of several forecasts. Three techniques are based on statistical methods: LR [128], SARIMA [142], and QR [151]; six techniques are based on machine learning: GP [150], SVM [146], deep residual network (DRN) [144], and three versions of AR-NARX method [128] based on linear regression (AR) [128], fixed size least squares SVM using the Nyström method (ARNFS) [161], and fixed size least squares SVM using Random Features (ARRFFS) [164]; finally, two techniques are based on weighted combination of several forecasts: secondary forecasting based on deviation analysis (SFDA) [148] and adaptive forgetting factor (AFF) [155].

In the first set of numerical results we quantify the prediction error of APLF in comparison with the 11 existing techniques for the 7 datasets. RMSE and MAPE assessing overall prediction errors are given in Table 4.1. Such table shows that existing techniques such as DRN and QR can achieve high accuracies in certain large-size regions using sizeable training datasets (e.g., New England dataset), however such techniques become inaccurate in other datasets such as those corresponding with small-size regions and smaller training datasets (e.g., 100 buildings dataset). Table 4.1 also shows that the online learning method AFF achieves higher accuracies than multiple offline learning algorithms such as LR, ARRFFS, and SFDA. Figures 4.6 and 4.7 provide more detailed comparisons using 5 representative existing techniques (AR, SARIMA, QR, SVM, and AFF) in comparison with proposed APLF in 3 datasets that correspond with regions of assorted sizes. Figure 4.6 shows two days of load demand and load forecasts in the three regions while Figure 4.7 shows the empirical cumulative distribution functions (CDFs) of the absolute value of prediction errors. Table 4.1, and Figures 4.6 and 4.7 show that the proposed APLF method achieves high accuracies in comparison with existing techniques in every dataset studied. In particular, Figure 4.7 shows that high errors occur with low probability for APLF method. For instance, in New England



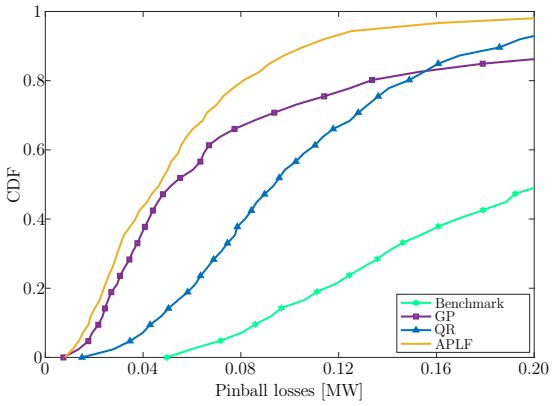
(a) New England. (b) Dayton. (c) 100 buildings.

Figure 4.8: Calibration plots describe the reliability of probabilistic forecasts of APLF method in comparison with state-of-the-art probabilistic techniques in three regions with different sizes.

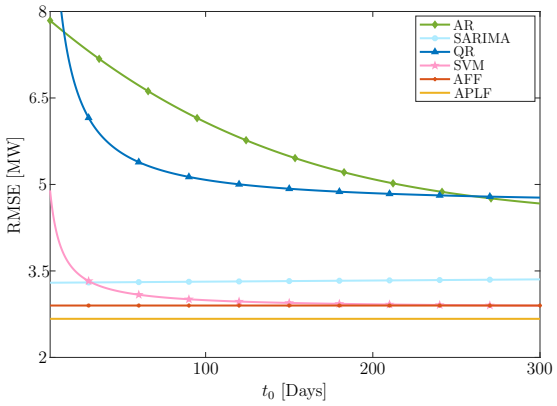
dataset, the error of APLF method is less than 0.8 GW with probability 0.8, while the 5 other methods reach errors of around 1.3 GW with such probability.

In the second set of numerical results we quantify the probabilistic performance of APLF in comparison with QR [151] and GP [150] and we study the relationship between training size and prediction error. Pinball loss and ECE assessing probabilistic forecasts are given in Table 4.2. Such table shows that APLF achieves high performance in terms of both pinball loss and ECE, while GP sometimes achieves poor results in terms of ECE and QR achieves poor results in terms of pinball loss. Figures 4.9a and 4.8 provide more detailled quantification of the probabilistic performance of different methods. Figure 4.9a shows the empirical CDFs of pinball losses of APLF method, QR, GP, and the benchmark for the GEFCom2014 dataset [156]. These CDFs show that the probability of high pinball losses is significantly lower for APLF method. In particular, the CDFs in Figure 4.9a show that APLF and GP have a similar median pinball loss of around 0.04 MW. However, APLF has pinball losses less than 0.08 MW with probability 0.8, while GP reaches pinball losses of 0.16 MW with such probability. Figure 4.8 shows the correspondence between the calibration $C(q)$ of probabilistic forecasts and the quantile q for the datasets used in Figures 4.6 and 4.7. These calibration plots show that GP and QR tend to obtain forecast quantiles higher than the true quantiles, while APLF obtains more unbiased probabilistic forecasts. In particular, the true load is higher than the 50 quantile forecast load with probability very near 50 % for APLF. In addition, Figure 4.8 shows that APLF obtains improved calibrations especially in the lower quantiles.

Finally, Figure 4.9b shows the RMSE obtained by APLF method and the 5 existing techniques shown in Figures 4.6 and 4.7 for different sizes of training sets using GEFCom2012 dataset. These results are obtained computing RMSEs 20 times for each size of training set. The samples used for training in these numerical results are different at each experiment and testing sets always contain two years of data. As can be observed from Figure 4.9b, the accuracy of online learning algorithms does not significantly change with the length of the training dataset, while offline learning algorithms



(a) CDFs of pinball losses compare the probabilistic performance of APLF method with state-of-the-art probabilistic techniques.



(b) The length of the training dataset does not affect the accuracy of online learning algorithms but significantly affects the accuracy of offline learning algorithms.

Figure 4.9: Results of the proposed APLF method in comparison with state-of-the-art techniques.

require large training datasets to achieve accurate results.

APLF method achieves remarkable results both in terms of single-value and probabilistic forecasts, and adapts to different consumption patterns in every region studied even where variability in load demand is more significant. Numerical results confirm that APLF better captures dynamic changes in consumption patterns than existing methods.

4.6 Conclusion

This chapter proposes techniques for adaptive probabilistic load forecasting (APLF) that can adapt to changes in consumption patterns and assess load uncertainties. We developed online learning techniques that update model parameters using a simple recursive algorithm, and prediction techniques that obtain probabilistic forecasts using the most recent parameters. In addition, we described the theoretical guarantees and efficient implementation of the online learning and probabilistic prediction steps for APLF. This chapter also compared the accuracy of the proposed APLF with existing techniques in multiple datasets. These datasets represent challenging scenarios with different sizes and different consumption patterns that change over time. The experimental results show the performance improvement of APLF method in terms of prediction errors and probabilistic forecasts. The proposed method can improve forecasting performance in a wide range of scenarios using efficient and flexible algorithms for adaptive online learning.

Chapter 5

Conclusions and future work

This section first briefly draws the conclusions of this dissertation and then, we describe the future extensions.

5.1 Conclusions

This dissertation proposes a learning methodology for time-dependent environments, techniques for supervised classification under concept drift (SCD), techniques for continual learning (CL), and learning and forecasting techniques for load forecasting. The proposed methodologies account for multidimensional tasks' changes, efficiently exploit information from different tasks in the sequence, and provide computable performance guarantees. In addition, the dissertation analytically characterizes the increase in effective sample size (ESS) achieved by the proposed methodology in terms of the expected quadratic change and the number of tasks. The numerical results assess the reliability of the performance guarantees presented and show the performance improvement in multiple machine learning scenarios using different datasets, sample sizes, and number of tasks.

Chapter 2 further describes the proposed methodology for SCD. Specifically, Chapter 2 proposes the methodology of adaptive minimax risk classifiers (AMRCs) that estimate multiple statistical characteristics of the time-varying underlying distribution and provide both qualitative and computable tight performance guarantees. The numerical results assess the performance improvement of AMRCs with respect to the state-of-the-art using benchmark datasets. The proposed methodology enables to more fully exploit data with characteristics that change over time and can provide more informative performance guarantees under concept drift.

Chapter 3 further describes the proposed methodology for CL. Specifically, Chapter 3 proposes continual learning methods based on minimax risk classifiers (CL-MRCs) that carefully avoid the repeated usage of the same information from the ever-increasing sequence of succeeding tasks and account for time-dependent tasks. In addition, we analytically characterize the increase in ESS achieved by the proposed forward and

backward learning techniques that show the positive backward transfer achieved by the proposed techniques. The numerical results assess the performance improvement of CL-MRC methodology with respect to existing methods using multiple datasets, sample sizes, and number of tasks. The proposed methodology for CL with time-dependent tasks can lead to techniques that further approach the humans' ability to learn from few examples and to continuously improve on tasks that arrive over time.

Chapter 4 proposes techniques for adaptive probabilistic load forecasting (APLF) that can adapt to changes in consumption patterns and assess load uncertainties. We developed online learning techniques that update model parameters using a simple recursive algorithm, and prediction techniques that obtain probabilistic forecasts using the most recent parameters. In addition, we described the theoretical guarantees and efficient implementation of the online learning and probabilistic prediction steps for APLF. The experimental results show the performance improvement of APLF method in terms of prediction errors and probabilistic forecasts. The proposed method can improve forecasting performance in a wide range of scenarios using efficient and flexible algorithms for adaptive online learning.

5.2 Future work

In this dissertation, we propose learning techniques for supervised classification setting in which the data available are time-dependent. This work can be extended to learning scenarios where data are weak classifier predictions and the weak classifiers arrive over time. Classification in those scenarios is commonly referred to as weak supervision or weak classification.

Supervised classification requires a large set of labeled data to learn classification rules. In many applications, getting labeled data is expensive and the limited amount of labeled data is not enough to learn accurate models. Weak supervision learns by efficiently combining the prediction of unlabeled data given by a set of weak classifiers obtained from related tasks.

The development of weak classification techniques is hindered by the quality and the quantity of weak classifiers. To address this problems, the first works on weak supervision commonly assume that the weak classifiers make errors independently with respect to the classification task of interest. However, in weak supervision the weak classifiers may not be independent. For instance, if we want to classify dogs in animal images and a weak classifier predicts 4-legged animals as dog and a weak classifier predicts terrestrial animals as dog; then using these weak classifiers a cat will be classified as a dog. Therefore, the weak classifiers have non-independent errors.

In weak supervision, let $\{\mathrm{h}_1, \mathrm{h}_2, \ldots, \mathrm{h}_N\}$ be an ensemble of N weak classifiers and $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ be a set of n unlabelled instances. Each classification rule in the ensemble h_j assigns label $y_i \in \mathcal{Y}$ to instance $x_i \in \mathcal{X}$ with probability $\mathrm{h}_j(y_i|x_i)$ for any $j = 1, 2, \ldots, N$ and $i = 1, 2, \ldots, n$. In addition, let $\Phi_j(x_i, y_i)$ be such that $\Phi_j(x_i, y_i) = 1 - \mathrm{h}_j(y_i|x_i)$ and e_j be an estimate of the error probability of the weak

classifier h_j for $j = 1, 2, \ldots, N$. Learning methods aim to obtain a classification rule h with small expected loss $\ell(\mathrm{h}, \mathrm{p})$ using the ensemble of weak classifiers and the set of instances.

As future work, we are going to develop learning techniques based on minimax risk classifiers (MRCs) that obtain probabilistic predictions using weak classifier predictions. In addition, the developed techniques will be able to incorporate new weak classifiers over time that will allow us to obtain more accurate predictions without access to the ground truth. We also want to obtain confidence of our probabilistic predictions, prove the consistency of the methods developed, and show the relationship with the consistency of the state-of-the-art methods.

Appendix A

Learning in Time-dependent Environments based on Minimax Classification

A.1 Derivation of recursions (1.5)-(1.6), (1.2)-(1.3), and (1.8)-(1.9)

This section shows how recursions (1.5), (1.6), recursions (1.2), (1.3), and recursions (1.8), (1.9) are obtained using those for filtering in linear dynamical systems.

The mean vectors evolve over tasks through the linear dynamical system

$$\boldsymbol{\tau}_j^{\infty} = \boldsymbol{\tau}_{j-1}^{\infty} + \boldsymbol{w}_j \tag{A.1}$$

where vectors $\boldsymbol{w}_j$ for $j \in \{2, 3, \ldots, k\}$ are independent and zero-mean. In addition, each state variable $\boldsymbol{\tau}_j^{\infty}$ is observed at each step j through $\boldsymbol{\tau}_j$ that is the sample average of i.i.d. samples from p_j, so that we have

$$\boldsymbol{\tau}_j = \boldsymbol{\tau}_j^{\infty} + \boldsymbol{v}_j \tag{A.2}$$

where $\boldsymbol{v}_j$ for $j \in \{1, 2, \ldots, k\}$ are independent and zero mean, and independent of $\boldsymbol{w}_j$ for $j \in \{1, 2, \ldots, k\}$. Therefore, equations (A.1) and (A.2) above describe a linear dynamical system (state-space model with white noise processes) [120, 165]. For such systems, the Kalman filter recursions provide the unbiased linear estimator with minimum MSE based on samples corresponding to preceding steps $D_1, D_2, \ldots, D_j$, and fixed-lag smoother recursions provide the unbiased linear estimator with minimum MSE based on samples corresponding to preceding and succeeding steps $D_1, D_2, \ldots, D_k$ [120, 165]. Then, equations (1.5), (1.6), equations (1.2), (1.3), and equations (1.8), (1.9) are obtained after some algebra from the Kalman filter recursions and fixed-lag smoother recursions.

A.2 Proof of Theorem 2

Proof. To obtain bound in (1.12) we first prove that the mean vector estimate and the MSE vector given by (1.2) and (1.3), respectively, satisfy

$$\mathbb{P}\left\{|\tau_j^{\infty(i)} - \tau_j^{j(i)}| \leq \kappa\sqrt{2s_j^{j(i)}\log\left(\frac{2m}{\delta}\right)}\right\} \geq (1-\delta) \tag{A.3}$$

for any component $i = 1, 2, \ldots, m$. Then, we prove that $\|\sqrt{\boldsymbol{s}_j^j}\|_\infty \leq M/\sqrt{n_j^j}$ for $j \in \{1, 2, \ldots, k\}$, where the ESSs satisfy $n_1^1 = n_1$ and $n_j^j \geq n_j + n_{j-1}^{j-1}\frac{\|\boldsymbol{\sigma}_j^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty + n_{j-1}^{j-1}\|\boldsymbol{d}_j^2\|_\infty}$ for $j \geq 2$.

To obtain inequality (A.3), we prove by induction that each component $i = 1, 2, \ldots, m$ of the error in the mean vector estimate $z_j^{j(i)} = \tau_j^{\infty(i)} - \tau_j^{j(i)}$ is sub-Gaussian with parameter $\eta_j^{j(i)} \leq \kappa\sqrt{s_j^{j(i)}}$. Firstly, for $j = 1$, we have that

$$z_1^{1(i)} = \tau_1^{\infty(i)} - \tau_1^{1(i)} = \tau_1^{\infty(i)} - \tau_1^{(i)}.$$

Since the bounded random variable $\Phi_1^{(i)}$ is sub-Gaussian with parameter $\sigma(\Phi_1^{(i)})$, then the error in the mean vector estimate $z_1^{1(i)}$ is sub-Gaussian with parameter that satisfies

$$\left(\eta_1^{1(i)}\right)^2 = \frac{\sigma\left(\Phi_1^{(i)}\right)^2}{n_1} \leq \frac{\kappa^2\sigma_1^{2(i)}}{n_1} = \kappa^2 s_1^{(i)}.$$

If $z_{j-1}^{j-1(i)} = \tau_{j-1}^{\infty}{}^{(i)} - \tau_{j-1}^{j-1(i)}$ is sub-Gaussian with parameter $\eta_{j-1}^{j-1(i)} \leq \kappa\sqrt{s_{j-1}^{j-1(i)}}$ for any $i = 1, 2, \ldots, m$, then using the recursions (1.2) and (1.3) we have that

$$\begin{aligned}
z_j^{j(i)} &= \tau_j^{\infty(i)} - \tau_j^{j(i)} \\
&= \tau_{j-1}^{\infty}{}^{(i)} + w_j^{(i)} - \tau_j^{(i)} - \frac{s_j^{(i)}}{s_{j-1}^{j-1(i)} + s_j^{(i)} + d_j^{2(i)}}\left(\tau_{j-1}^{j-1(i)} - \tau_j^{(i)}\right) \\
&= \tau_{j-1}^{\infty}{}^{(i)} + w_j^{(i)} - \tau_{j-1}^{j-1(i)} + \left(1 - \frac{s_j^{(i)}}{s_{j-1}^{j-1(i)} + s_j^{(i)} + d_j^{2(i)}}\right)\left(\tau_{j-1}^{j-1(i)} - \tau_j^{(i)}\right) \\
&= \tau_{j-1}^{\infty}{}^{(i)} + w_j^{(i)} - \tau_{j-1}^{j-1(i)} - \frac{s_j^{j(i)}}{s_j^{(i)}}\left(\tau_j^{(i)} - \tau_{j-1}^{j-1(i)}\right)
\end{aligned}$$

since $\boldsymbol{w}_j = \boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_{j-1}^\infty$. If $\boldsymbol{v}_j = \boldsymbol{\tau}_j - \boldsymbol{\tau}_j^\infty$, the error in the mean vector estimate is given

by

$$\begin{aligned} z_j^{j(i)} &= \tau_{j-1}^{\infty}{}^{(i)} + w_j^{(i)} - \tau_{j-1}^{j-1(i)} - \frac{s_j^{j(i)}}{s_j^{(i)}} \left(\tau_j^{\infty(i)} + v_j^{(i)} - \tau_{j-1}^{j-1(i)} \right) \\ &= \tau_{j-1}^{\infty}{}^{(i)} + w_j^{(i)} - \tau_{j-1}^{j-1(i)} - \frac{s_j^{j(i)}}{s_j^{(i)}} \left(\tau_{j-1}^{\infty}{}^{(i)} + w_j^{(i)} + v_j^{(i)} - \tau_{j-1}^{j-1(i)} \right) \\ &= \left(1 - \frac{s_j^{j(i)}}{s_j^{(i)}} \right) z_{j-1}^{j-1(i)} + \left(1 - \frac{s_j^{j(i)}}{s_j^{(i)}} \right) \left(w_j^{(i)} \right) - \frac{s_j^{j(i)}}{s_j^{(i)}} v_j^{(i)} \end{aligned}$$

where $w_j^{(i)}$ and $v_j^{(i)}$ are sub-Gaussian with parameter $\sigma(w_j^{(i)})$ and $\sigma\left(\Phi_j^{(i)}\right)/\sqrt{n_j}$, respectively. Therefore, we have that $z_j^{j(i)}$ is sub-Gaussian with parameter $\eta_j^{j(i)}$ that satisfies

$$\begin{aligned} \left(\eta_j^{j(i)} \right)^2 = &\left(1 - \frac{s_j^{j(i)}}{s_j^{(i)}} \right)^2 \left(\eta_{j-1}^{j-1(i)} \right)^2 \\ &+ \left(1 - \frac{s_j^{j(i)}}{s_j^{(i)}} \right)^2 \sigma \left(w_j^{(i)} \right)^2 + \left(\frac{s_j^{j(i)}}{s_j^{(i)}} \right)^2 \frac{\sigma \left(\Phi_j^{(i)} \right)^2}{n_j} \end{aligned} \tag{A.4}$$

since $\boldsymbol{z}_{j-1}^{j-1}$, $\boldsymbol{w}_j$, and $\boldsymbol{v}_j$ are independent. Using that $\eta_{j-1}^{j-1(i)} \leq \kappa \sqrt{s_{j-1}^{j-1(i)}}$ and the definition of κ, we have that

$$\begin{aligned} \left(\eta_j^{j(i)} \right)^2 &\leq \left(1 - \frac{s_j^{j(i)}}{s_j^{(i)}} \right)^2 \kappa^2 s_{j-1}^{j-1(i)} + \left(1 - \frac{s_j^{j(i)}}{s_j^{(i)}} \right)^2 \kappa^2 d_j^{2(i)} \\ &\quad + \left(\frac{s_j^{j(i)}}{s_j^{(i)}} \right)^2 \kappa^2 \frac{\sigma_j^{2(i)}}{n_j} \end{aligned} \tag{A.5}$$

$$\begin{aligned} &\leq \left(1 - \frac{s_j^{j(i)}}{s_j^{(i)}} \right)^2 \kappa^2 \left(\left(\frac{1}{s_j^{j(i)}} - \frac{1}{s_j{}^{(i)}} \right)^{-1} + d_j^{2(i)} \right) + \frac{\left(s_j^{j(i)} \right)^2}{s_j^{(i)}} \kappa^2 \\ &= \left(1 - \frac{s_j^{j(i)}}{s_j^{(i)}} \right) \kappa^2 s_j^{j(i)} + \kappa^2 \frac{\left(s_j^{j(i)} \right)^2}{s_j^{(i)}} \end{aligned} \tag{A.6}$$

where (A.6) is obtained using (3.2).

The inequality in (A.3) is obtained using the union bound together with the Chernoff bound (concentration inequality) [85] for the random variables $z_j^{j(i)}$ that are sub-Gaussian with parameter $\eta_j^{j(i)}$.

Now, we prove by induction that, for any j, $\|\sqrt{\boldsymbol{s}_j^j}\|_\infty \leq M/\sqrt{n_j^j}$ where the ESSs satisfy $n_1^1 = n_1$ and $n_j^j \geq n_j + n_{j-1}^{j-1}\frac{\|\boldsymbol{\sigma}_j^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty + n_{j-1}^{j-1}\|\boldsymbol{d}_j^2\|_\infty}$ for $j \geq 2$. For $j = 1$, using the definition of $\boldsymbol{s}_j^j$ in equation (3.2), we have that for any component i

$$\left(s_1^{1(i)}\right)^{-1} = \left(s_1^{(i)}\right)^{-1} = \frac{n_1}{\sigma_1^{2(i)}} \geq \frac{n_1}{M^2}.$$

Then, vector $\boldsymbol{s}_1^1$ satisfies

$$\|\sqrt{\boldsymbol{s}_1^1}\|_\infty \leq \frac{M}{\sqrt{n_1}} = \frac{M}{\sqrt{n_1^1}}.$$

If $\|\sqrt{\boldsymbol{s}_{j-1}^{j-1}}\|_\infty \leq M/\sqrt{n_{j-1}^{j-1}}$, then we have that for any component i

$$\begin{aligned}\left(s_j^{j(i)}\right)^{-1} &= \frac{1}{s_j^{(i)}} + \frac{1}{s_{j-1}^{j-1(i)} + d_j^{2(i)}} \geq \frac{1}{s_j^{(i)}} + \frac{1}{\frac{M^2}{n_{j-1}^{j-1}} + d_j^{2(i)}} \\ &\geq \frac{1}{M^2}\left(n_j + \frac{1}{\frac{1}{n_{j-1}^{j-1}} + \frac{d_j^{2(i)}}{M^2}}\right) \geq \frac{1}{M^2}\left(n_j + \frac{1}{\frac{1}{n_{j-1}^{j-1}} + \frac{\|\boldsymbol{d}_j^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty}}\right)\end{aligned}$$

by using the recursion (3.2) and the induction hypothesis. Then, vector $\boldsymbol{s}_j^j$ satisfies

$$\left\|\sqrt{\boldsymbol{s}_j^j}\right\|_\infty \leq \frac{M}{\sqrt{n_j + n_{j-1}^{j-1}\frac{\|\boldsymbol{\sigma}_j^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty + n_{j-1}^{j-1}\|\boldsymbol{d}_j^2\|_\infty}}}. \tag{A.7}$$

The inequality in (3.4) is obtained because the minimax risk is bounded by the smallest minimax risk as shown in [76–78] so that

$$R(\mathcal{U}_j^j) \leq R_j^\infty + \left(\|\boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_j^j\|_\infty + \|\boldsymbol{\lambda}_j^j\|_\infty\right)\left\|\boldsymbol{\mu}_j^\infty\right\|_1 \tag{A.8}$$

that leads to (3.4) using (A.3), (A.7), and the fact that $1 \leq \sqrt{2\log\left(\frac{2m}{\delta}\right)}$. □

A.3 Proof of Theorem 3

Proof. To obtain bound in (1.14), we proceed by induction. For $j = 1$, using the expression for the ESS in (1.13), we have that

$$n_1^1 = n_1 \geq n.$$

If (1.14) holds for the $(j-1)$-task, then for the j-th task, we have that

$$\begin{aligned} n_j^j &\geq n_j + n_{j-1}^{j-1} \frac{\|\boldsymbol{\sigma}_j^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty + n_{j-1}^{j-1}\|\boldsymbol{d}_j^2\|_\infty} \\ &\geq n + n_{j-1}^{j-1} \frac{1}{1 + n_{j-1}^{j-1} d^2} = n\left(1 + \frac{1}{\frac{n}{n_{j-1}^{j-1}} + nd^2}\right) \end{aligned}$$

where the second inequality is obtained because $n_j \geq n$, $\|\boldsymbol{\sigma}_j^2\|_\infty \leq 1$, and $\|\boldsymbol{d}_j^2\|_\infty \leq d^2$. Using that $n_{j-1}^{j-1} \geq n\left(1 + \frac{(1+\alpha)^{2j-3}-1-\alpha}{\alpha(1+\alpha)^{2j-3}+\alpha}\right)$, the ESS of the j-th task satisfies

$$\begin{aligned} n_j^j &\geq n\left(1 + \frac{1}{\frac{n}{n\left(1+\frac{(1+\alpha)^{2j-3}-1-\alpha}{\alpha(1+\alpha)^{2j-3}+\alpha}\right)} + nd^2}\right) = n\left(1 + \frac{1}{\frac{\alpha(1+\alpha)^{2j-3}+\alpha}{(1+\alpha)^{2j-2}-1} + nd^2}\right) \\ &= n\left(1 + \frac{1}{\frac{\alpha(1+\alpha)^{2j-3}+\alpha}{(1+\alpha)^{2j-2}-1} + \frac{\alpha^2}{\alpha+1}}\right) \qquad \text{(A.9)} \\ &= n\left(1 + \frac{(1+\alpha)^{2j-1}-1-\alpha}{\alpha(1+\alpha)^{2j-2} + \alpha(\alpha+1) + \alpha^2(1+\alpha)^{2j-2} - \alpha^2}\right) \end{aligned}$$

where (A.9) is obtained because $nd^2 = \frac{\alpha^2}{\alpha+1}$ since $\alpha = \frac{nd^2}{2}\left(\sqrt{1+\frac{4}{nd^2}}+1\right)$.

Now, we obtain bounds for the ESS depending on the value of nd^2. In the following, the constant ϕ represents the golden ratio $\phi = 1.618\ldots$.

1. If $nd^2 < \frac{1}{j^2} \Rightarrow \sqrt{nd^2} \leq \alpha \leq \sqrt{nd^2}\phi \leq \frac{\phi}{j} \leq 1$ similarly as in the previous case, then we have that n_j^j satisfies

   $$n_j^j \geq n\left(1 + \frac{1}{\alpha}\frac{\alpha(2j-2)}{2+\alpha(2j-1)}\right) = n\left(1 + \frac{2j-2}{2+\alpha(2j-1)}\right)$$

   where the first inequality follows because $(1+\alpha)^{2j-2} \geq 1 + \alpha(2j-2)$. Using $\alpha \leq \frac{\phi}{j}$, we have that

   $$\begin{aligned} n_j^j &\geq n\left(1 + \frac{2j-2}{2+\frac{\phi}{j}(2j-1)}\right) \\ &\geq n\left(1 + \frac{2j-2}{2+2\phi-\frac{\phi}{j}}\right) \geq n\left(1 + \frac{j-1}{1+\phi}\right). \end{aligned}$$

2. If $\frac{1}{j^2} \leq nd^2 < 1 \Rightarrow \frac{1}{j} < \sqrt{nd^2} < \alpha < \sqrt{nd^2}\phi$ because

   $$\alpha = \frac{nd^2}{2}\left(\sqrt{1+\frac{4}{nd^2}}+1\right) = \sqrt{nd^2}\frac{\sqrt{nd^2+4}+\sqrt{nd^2}}{2}$$

then we have that n_j^j satisfies

$$n_j^j \geq n\left(1+\frac{1}{\sqrt{nd^2}}\frac{1}{\phi}\frac{(1+\alpha)^{2j-2}-1}{(1+\alpha)^{2j-2}+1}\right)$$
$$\geq n\left(1+\frac{1}{\sqrt{nd^2}}\frac{1}{\phi}\frac{(1+\frac{1}{j})^{2j-2}-1}{(1+\frac{1}{j})^{2j-2}+1}\right)$$

where the first inequality follows because $\alpha < \sqrt{nd^2}\phi$ and the second inequality follows because the expression is monotonically increasing for α and $\frac{1}{j} < \alpha$. Since $(1+\frac{1}{j})^{2j-2} \geq 1+\frac{2j-2}{j}$, we have that

$$n_j^j \geq n\left(1+\frac{1}{\sqrt{nd^2}}\frac{1}{\phi}\frac{\frac{2j-2}{j}}{2+\frac{2j-2}{j}}\right) \geq n\left(1+\frac{1}{\sqrt{nd^2}}\frac{1}{\phi}\frac{1}{3}\right)$$

because $j \geq 2$.

3. If $nd^2 \geq 1 \;\Rightarrow 1 \leq nd^2 \leq \alpha \leq nd^2\phi$ because $\alpha = \frac{nd^2}{2}\left(\sqrt{1+\frac{4}{nd^2}}+1\right)$, then we have that n_j^j satisfies

$$n_j^j \geq n\left(1+\frac{1}{nd^2\phi}\frac{(1+\alpha)^{2j-1}-1-\alpha}{(1+\alpha)^{2j-1}+1}\right)$$
$$\geq n\left(1+\frac{1}{nd^2\phi}\frac{(1+\alpha)^{2j-2}-1}{(1+\alpha)^{2j-2}+1}\right)$$

where the first inequality follows because $\alpha \leq nd^2\phi$ and the second inequality follows multiplying and dividing by $1+\alpha$ and because $1/(1+\alpha) < 1$. Since the above expression is monotonically increasing for α and $\alpha \geq 1$, we have that

$$n_j^j \geq n\left(1+\frac{1}{nd^2\phi}\frac{2^{2j-2}-1}{2^{2j-2}+1}\right) \geq n\left(1+\frac{1}{nd^2\phi}\frac{3}{5}\right)$$

because $j \geq 2$.

□

A.4 Proof of bounds for mean and mean squared error (MSE) vectors obtained as in (1.5)-(1.6)

Proof. To obtain bound in (1.12) with $R(\mathcal{U}_j^{j-1})$ and n_j^{j-1} instead of $R(\mathcal{U}_j^j)$ and n_j^j, we first prove that the mean vector estimate and the MSE vector given by (1.5) and (1.6), respectively, satisfy

$$\mathbb{P}\left\{|\tau_j^{\infty(i)} - \tau_j^{j-1(i)}| \leq \kappa\sqrt{2s_j^{j-1(i)}\log\left(\frac{2m}{\delta}\right)}\right\} \geq (1-\delta) \tag{A.10}$$

for any component $i = 1, 2, \ldots, m$ and that $\|\sqrt{\boldsymbol{s}_j^{j-1}}\|_\infty \leq M/\sqrt{n_j^{j-1}}$ for $j \in \{1, 2, \ldots, k\}$, where the ESS satisfies $n_2^1 = \|\boldsymbol{\sigma}_1^2\|_\infty / \|\boldsymbol{d}_2\|_\infty$ and

$$n_j^{j-1} \geq (n_{j-1} + n_{j-1}^{j-2}) \frac{\|\boldsymbol{\sigma}_{j-1}^2\|_\infty}{(n_{j-1} + n_{j-1}^{j-2})\|\boldsymbol{d}_j\|_\infty + \|\boldsymbol{\sigma}_{j-1}^2\|_\infty}$$

for $j \geq 3$.

To obtain inequality (A.10), we prove by induction that each component $i = 1, 2, \ldots, m$ of the error in the mean vector estimate $z_j^{j-1(i)} = \tau_j^{\infty(i)} - \tau_j^{j-1(i)}$ is sub-Gaussian with parameter $\eta_j^{j-1(i)} \leq \kappa\sqrt{s_j^{j-1(i)}}$. Firstly, for $j = 2$, we have that

$$z_2^{1(i)} = \tau_2^{\infty(i)} - \tau_2^{1(i)} = \tau_2^{\infty(i)} - \tau_1^{(i)}.$$

Since the bounded random variable $\Phi_1^{(i)}$ is sub-Gaussian with parameter $\sigma(\Phi_1^{(i)})$, then the error in the mean vector estimate $z_2^{1(i)}$ is sub-Gaussian with parameter that satisfies

$$\left(\eta_2^{1(i)}\right)^2 = \frac{\sigma\left(\Phi_1^{(i)}\right)^2}{n_1} \leq \frac{\kappa^2 \sigma_1^{2(i)}}{n_1} = \kappa^2 s_1^{(i)}.$$

If $z_{j-1}^{j-2(i)} = \tau_{j-1}^{\infty}{}^{(i)} - \tau_{j-1}^{j-2(i)}$ is sub-Gaussian with parameter $\eta_{j-1}^{j-2(i)} \leq \kappa\sqrt{s_{j-1}^{j-2(i)}}$ for any $i = 1, 2, \ldots, m$, then using the recursions (1.5) and (1.6) we have that

$$\begin{aligned} z_j^{j-1(i)} &= \tau_j^{\infty(i)} - \tau_j^{j-1(i)} = \tau_{j-1}^{\infty}{}^{(i)} + w_j^{(i)} - \tau_{j-1}^{j-2(i)} - \eta_j^{(i)}(\tau_{j-1}^{(i)} - \tau_{j-1}^{j-2(i)}) \\ &= \tau_{j-1}^{\infty}{}^{(i)} + w_j^{(i)} - \tau_{j-1}^{j-2(i)} - \frac{n_{j-1} s_{j-1}^{j-2(i)}}{n_{j-1} s_{j-1}^{j-2(i)} + \sigma_{j-1}^{2}{}^{(i)}}(\tau_{j-1}^{(i)} - \tau_{j-1}^{j-2(i)}) \end{aligned}$$

since $\boldsymbol{w}_j = \boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_{j-1}^\infty$. If $\boldsymbol{v}_j = \boldsymbol{\tau}_j - \boldsymbol{\tau}_j^\infty$, the error in the mean vector estimate is given by

$$\begin{aligned} z_j^{j-1(i)} =& \tau_{j-1}^{\infty}{}^{(i)} + w_j^{(i)} - \tau_{j-1}^{j-2(i)} \qquad \text{(A.11)} \\ &- \frac{n_{j-1} s_{j-1}^{j-2(i)}}{n_{j-1} s_{j-1}^{j-2(i)} + \sigma_{j-1}^{2}{}^{(i)}} \left(\tau_{j-1}^{\infty}{}^{(i)} + v_{j-1}^{(i)} - \tau_{j-1}^{j-2(i)}\right) \\ =& \left(1 - \frac{n_{j-1} s_{j-1}^{j-2(i)}}{n_{j-1} s_{j-1}^{j-2(i)} + \sigma_{j-1}^{2}{}^{(i)}}\right) z_{j-1}^{j-2(i)} + w_j^{(i)} \qquad \text{(A.12)} \\ &- \frac{n_{j-1} s_{j-1}^{j-2(i)}}{n_{j-1} s_{j-1}^{j-2(i)} + \sigma_{j-1}^{2}{}^{(i)}} v_{j-1}^{(i)} \end{aligned}$$

where $w_j^{(i)}$ and $v_j^{(i)}$ are sub-Gaussian with parameter $\sigma(w_j^{(i)})$ and $\sigma\left(\Phi_j^{(i)}\right)/\sqrt{n_j}$, respectively. Therefore, we have that $z_j^{j-1(i)}$ is sub-Gaussian with parameter $\eta_j^{j-1(i)}$ that satisfies

$$\left(\eta_j^{j-1(i)}\right)^2 = \left(1 - \frac{n_{j-1}s_{j-1}^{j-2(i)}}{n_{j-1}s_{j-1}^{j-2(i)} + \sigma_{j-1}^{2\ (i)}}\right)^2 \left(\eta_{j-1}^{j-2(i)}\right)^2 + \sigma\left(w_j^{(i)}\right)^2 + \left(\frac{n_{j-1}s_{j-1}^{j-2(i)}}{n_{j-1}s_{j-1}^{j-2(i)} + \sigma_{j-1}^{2\ (i)}}\right)^2 \frac{\sigma\left(\Phi_{j-1}^{(i)}\right)^2}{n_{j-1}} \tag{A.13}$$

since $\boldsymbol{z}_{j-1}^{j-2}$, $\boldsymbol{w}_j$, and $\boldsymbol{v}_j$ are independent. Using that $\eta_{j-1}^{j-2(i)} \leq \kappa\sqrt{s_{j-1}^{j-2(i)}}$ and the definition of κ, we have that

$$\left(\eta_j^{j-1(i)}\right)^2 \leq \left(1 - \frac{n_{j-1}s_{j-1}^{j-2(i)}}{n_{j-1}s_{j-1}^{j-2(i)} + \sigma_{j-1}^{2\ (i)}}\right)^2 \kappa^2 s_{j-1}^{j-2(i)} + \kappa^2 d_j^{2(i)} + \left(\frac{n_{j-1}s_{j-1}^{j-2(i)}}{n_{j-1}s_{j-1}^{j-2(i)} + \sigma_{j-1}^{2\ (i)}}\right)^2 \kappa^2 \frac{\sigma_{j-1}^{2\ (i)}}{n_{j-1}} \tag{A.14}$$

$$\leq \left(1 - \frac{s_{j-1}^{j-2(i)}}{s_{j-1}^{j-2(i)} + s_{j-1}{}^{(i)}}\right) \kappa^2 (s_j^{j-1(i)} - d_j^{2(i)}) + \kappa^2 d_j^{2(i)} + \left(\frac{s_{j-1}^{j-2(i)}}{s_{j-1}^{j-2(i)} + s_{j-1}^{(i)}}\right)^2 \kappa^2 s_{j-1}^{(i)} \tag{A.15}$$

$$\leq \kappa^2 s_j^{j-1(i)} + \frac{s_{j-1}^{j-2(i)}}{s_{j-1}^{j-2(i)} + s_{j-1}^{(i)}} \kappa^2 \left(d_j^{2(i)} + \frac{s_{j-1}^{j-2(i)}}{s_{j-1}^{j-2(i)} + s_{j-1}^{(i)}} s_{j-1}^{(i)} - s_j^{j-1(i)}\right) \tag{A.16}$$

where (A.16) is obtained using (1.6).

The inequality in (A.10) is obtained using the union bound together with the Chernoff bound (concentration inequality) [85] for the random variables $z_j^{j-1(i)}$ that are sub-Gaussian with parameter $\eta_j^{j-1(i)}$.

Now, we prove by induction that, for any k, $\|\sqrt{\boldsymbol{s}_j^{j-1}}\|_\infty \leq M/\sqrt{n_j^{j-1}}$ where the ESSs satisfy $n_2^1 = \|\boldsymbol{\sigma}_1^2\|_\infty / \|\boldsymbol{d}_2\|_\infty$ and

$$n_j^{j-1} \geq (n_{j-1} + n_{j-1}^{j-2}) \frac{\|\boldsymbol{\sigma}_{j-1}^2\|_\infty}{(n_{j-1} + n_{j-1}^{j-2})\|\boldsymbol{d}_j\|_\infty + \|\boldsymbol{\sigma}_{j-1}^2\|_\infty}$$

for $j \geq 3$. For $j = 2$, using the definition of $\boldsymbol{s}_j^{j-1}$ in equation (1.6), we have that for any component i

$$\left(s_2^{1(i)}\right)^{-1} = \frac{1}{\boldsymbol{d}_2^{(i)}} \geq \frac{n_2^1}{M^2}.$$

Then, vector $\boldsymbol{s}_2^1$ satisfies

$$\|\sqrt{\boldsymbol{s}_2^1}\|_\infty \leq \frac{M}{\sqrt{n_2^1}}.$$

If $\|\sqrt{\boldsymbol{s}_{j-1}^{j-2}}\|_\infty \leq M/\sqrt{n_{j-1}^{j-2}}$, then we have that for any component i

$$\begin{aligned}
\left(s_j^{j-1(i)}\right)^{-1} &= \frac{1}{\left(1 - \frac{s_{j-1}^{j-2(i)}}{s_{j-1}^{(i)} + s_{j-1}^{j-2(i)}}\right) s_{j-1}^{j-2(i)} + d_j^{2(i)}} \geq \frac{1}{\frac{s_{j-1}^{(i)}}{s_{j-1}^{(i)} + \frac{M}{n_{j-1}^{j-2}}} \frac{M}{n_{j-1}^{j-2}} + d_j^{2(i)}} \\
&= \frac{1}{\frac{M s_{j-1}^{(i)}}{s_{j-1}^{(i)} n_{j-1}^{j-2} + M} + d_j^{2(i)}} = \frac{1}{\frac{\frac{M \sigma_{j-1}^{2}{}^{(i)}}{n_{j-1}}}{\frac{\sigma_{j-1}^{2}{}^{(i)}}{n_{j-1}} n_{j-1}^{j-2} + M} + d_j^{2(i)}} \\
&= \frac{\sigma_{j-1}^{2}{}^{(i)} n_{j-1}^{j-2} + M n_{k-1}}{M \sigma_{k-1}^{2}{}^{(i)} + (\sigma_{j-1}^{2}{}^{(i)} n_{j-1}^{j-2} + M n_{j-1}) d_j^{2(i)}} \\
&\geq \frac{n_{j-1}^{j-2} + n_{j-1}}{M + (n_{j-1}^{j-2} + n_{j-1}) d_j^{2(i)}}
\end{aligned}$$

by using the recursion (1.6) and the induction hypothesis. Then, vector $\boldsymbol{s}_j^{j-1}$ satisfies

$$\left\|\sqrt{\boldsymbol{s}_j^{j-1}}\right\|_\infty \leq \frac{M}{\sqrt{(n_{j-1} + n_{j-1}^{j-2}) \frac{\|\boldsymbol{\sigma}_{j-1}^2\|_\infty}{(n_{j-1} + n_{j-1}^{j-2}) \|\boldsymbol{d}_j\|_\infty + \|\boldsymbol{\sigma}_{j-1}^2\|_\infty}}} \leq \frac{M}{\sqrt{n_j^{j-1}}}$$

with $n_j^{j-1} \geq (n_{j-1} + n_{j-1}^{j-2}) \frac{\|\boldsymbol{\sigma}_{j-1}^2\|_\infty}{(n_{j-1} + n_{j-1}^{j-2}) \|\boldsymbol{d}_j\|_\infty + \|\boldsymbol{\sigma}_{j-1}^2\|_\infty}$.

The inequality in (1.12) with $R(\mathcal{U}_j^{j-1})$ and n_j^{j-1} instead of $R(\mathcal{U}_j^j)$ and n_j^j is obtained because the minimax risk is bounded by the smallest minimax risk as shown in [76–78] so that we have (A.8) that leads to (1.12) with $R(\mathcal{U}_j^{j-1})$ and n_j^{j-1} instead of $R(\mathcal{U}_j^j)$ and n_j^j using (A.3), (A.7), and the fact that $1 \leq \sqrt{2 \log\left(\frac{2m}{\delta}\right)}$. □

A.5 Proof of Theorem 3

Proof. To obtain bound in (1.14), we proceed by induction. For $j = 1$, using the expression for the ESS in (1.12), we have that

$$n_1^0 = 0$$

If (1.14) holds for the $(j-1)$-task, then for the j-th task, we have that

$$n_j^{j-1} \geq (n_{j-1} + n_{j-1}^{j-2}) \frac{\|\boldsymbol{\sigma}_{j-1}^2\|_\infty}{\|\boldsymbol{\sigma}_{j-1}^2\|_\infty + \|\boldsymbol{d}_j\|_\infty (n_{j-1} + n_{j-1}^{j-2})} \geq \frac{n + n_{j-1}^{j-2}}{(n + n_{j-1}^{j-2})d^2 + 1}$$

where the second inequality is obtained because $n_{j-1} \geq n$, $\|\boldsymbol{\sigma}_j^2\|_\infty \leq 1$, and $\|\boldsymbol{d}_j\|_\infty \leq d$. Using that $n_{j-1}^{j-2} \geq n \frac{(1+\alpha)^{2j-3}-1-\alpha}{\alpha(1+\alpha)^{2j-3}+\alpha}$, the ESS of the j-th task satisfies

$$\begin{aligned} n_j^{j-1} &\geq n \frac{(1+\alpha)^{2j-2} - 1}{((1+\alpha)^{2j-2} - 1)nd^2 + \alpha(1+\alpha)^{2j-3} + \alpha} \\ &= n \frac{(1+\alpha)^{2j-1} - 1 - \alpha}{\alpha^2(1+\alpha)^{2j-2} - \alpha^2 + \alpha(1+\alpha)^{2j-2} + \alpha(\alpha+1)} \qquad \text{(A.17)} \\ &= n \frac{(1+\alpha)^{2j-1} - 1 - \alpha}{\alpha(1+\alpha)^{2j-1} + \alpha} \end{aligned}$$

where (A.17) is obtained because $nd = \frac{\alpha^2}{\alpha+1}$ since $\alpha = \frac{nd}{2}\left(\sqrt{1 + \frac{4}{nd}} + 1\right)$. □

A.6 Proof of Theorem 4

Proof. To obtain bound in (1.17) we first prove that the mean vector estimate and the MSE vector given by (1.8) and (1.9), respectively, satisfy

$$\mathbb{P}\left\{ |\tau_k^{\infty(i)} - \tau_j^{k(i)}| \leq \kappa \sqrt{2 s_j^{k(i)} \log\left(\frac{2m}{\delta}\right)} \right\} \geq (1-\delta) \qquad \text{(A.18)}$$

for any component $i = 1, 2, \ldots, m$. Then, we prove that $\|\boldsymbol{s}_j^k\|_\infty \leq M/\sqrt{n_j^k}$ for $j \in \{1, 2, \ldots, k\}$, where the ESSs satisfy

$$n_j^k \geq \frac{\left(\|\boldsymbol{\sigma}_j^2\|_\infty + n_j^j d_{j+1}^{(i)}\right)^2 n_{j+1}^k}{\|\boldsymbol{\sigma}_j^2\|_\infty^2 + d_{j+1}^{(i)}(\|\boldsymbol{\sigma}_j^2\|_\infty + n_j^j d_{j+1}^{(i)}) n_{j+1}^k}$$

To obtain inequality (A.18), we prove by induction that each component $i = 1, 2, \ldots, m$ of the error in the mean vector estimate $z_j^{k(i)} = \tau_j^{\infty(i)} - \tau_j^{k(i)}$ is sub-Gaussian with parameter $\eta_j^{k(i)} \leq \kappa \sqrt{s_j^{k(i)}}$. Firstly, for $j = k$, using the proof of Theorem 2 in Appendix A.2 we have that $z_k^{k(i)} = \tau_k^{\infty(i)} - \tau_k^{k(i)}$ is sub-Gaussian with parameter that satisfies $\eta_k^{k(i)} \leq \kappa^2 s_k^{k(i)}$. If ${z_{j+1}^k}^{(i)} = {\tau_{j+1}^\infty}^{(i)} - {\tau_{j+1}^k}^{(i)}$ is sub-Gaussian with parameter ${\eta_{j+1}^k}^{(i)} \leq \kappa^2 {s_{j+1}^k}^{(i)}$ for any $i = 1, 2, \ldots, m$, then uthe error in the forward and backward

mean vector estimate is given by

$$\begin{aligned} z_j^{k(i)} &= \tau_j^{\infty(i)} - \tau_j^{k(i)} \\ &= \tau_j^{\infty(i)} - \tau_{j+1}^{k\ (i)} - \frac{d_{j+1}^{(i)}}{s_j^{j(i)} + d_{j+1}^{(i)}} \left(\tau_j^{j(i)} - \tau_{j+1}^{k\ (i)} \right) \end{aligned}$$

where the second equality is obtained using the recursion for $\tau_j^{k(i)}$ in (1.8). Adding and subtracting $\frac{d_{j+1}^{(i)}}{s_j^{j(i)} + d_{j+1}^{(i)}} \tau_j^{\infty(i)}$, we have that

$$\begin{aligned} z_j^{k(i)} &= \tau_j^{\infty(i)} - \tau_{j+1}^{k\ (i)} - \frac{d_{j+1}^{(i)}}{s_j^{j(i)} + d_{j+1}^{(i)}} \left(\tau_j^{\infty(i)} - \tau_j^{\infty(i)} + \tau_j^{j(i)} - \tau_{j+1}^{k\ (i)} \right) \\ &= \tau_j^{\infty(i)} - \tau_{j+1}^{k\ (i)} - \frac{d_{j+1}^{(i)}}{s_j^{j(i)} + d_{j+1}^{(i)}} \left(\tau_j^{\infty(i)} - z_j^{j(i)} - \tau_{j+1}^{k\ (i)} \right) \\ &= z_{j+1}^{k\ (i)} - w_{j+1}^{(i)} - \frac{d_{j+1}^{(i)}}{s_j^{j(i)} + d_{j+1}^{(i)}} \left(\tau_j^{\infty(i)} - z_j^{j(i)} - \tau_{j+1}^{k\ (i)} \right) \\ &= z_{j+1}^{k\ (i)} - w_{j+1}^{(i)} - \frac{d_{j+1}^{(i)}}{s_j^{j(i)} + d_{j+1}^{(i)}} \left(\tau_{j+1}^{\infty\ (i)} - w_{j+1}^{(i)} - z_j^{j(i)} - \tau_{j+1}^{k\ (i)} \right) \\ &= z_{j+1}^{k\ (i)} - w_{j+1}^{(i)} - \frac{d_{j+1}^{(i)}}{s_j^{j(i)} + d_{j+1}^{(i)}} \left(z_{j+1}^{k\ (i)} - w_{j+1}^{(i)} - z_j^{j(i)} \right) \end{aligned} \tag{A.19}$$

since $\boldsymbol{w}_j = \boldsymbol{\tau}_j^{\infty} - \boldsymbol{\tau}_{j-1}^{\infty}$ and $z_j^{j(i)} = \tau_j^{\infty(i)} - \tau_j^{j(i)}$ where $z_j^{j(i)}$, $z_{j+1}^{k\ (i)}$, and $w_{j+1}^{(i)}$ are sub-Gaussian with parameters $\eta_j^{j(i)} \leq \kappa \sqrt{s_j^{j(i)}}$, $\eta_{j+1}^{k\ (i)} \leq \kappa \sqrt{s_{j+1}^{k\ (i)}}$, and $\sigma(w_j^{(i)}) \leq \kappa \sqrt{d_j^{(i)}}$, respectively. Since $\boldsymbol{z}_j^j, \boldsymbol{z}_{j+1}^k$ and $\boldsymbol{w}_{j+1}$ are independent, we have that $z_j^{k(i)}$ given by (A.19)

is sub-Gaussian with parameter that satisfies

$$\begin{aligned}\left(\eta_j^{k(i)}\right)^2 &= \left(1-\frac{d_{j+1}^{(i)}}{s_j^{j(i)}+d_{j+1}^{(i)}}\right)^2 {\eta_{j+1}^k}^{(i)} + \left(\frac{d_{j+1}^{(i)}}{s_j^{j(i)}+d_{j+1}^{(i)}}-1\right)^2 \sigma\left(w_{j+1}^{(i)}\right)^2 \\ &\quad + \left(\frac{d_{j+1}^{(i)}}{s_j^{j(i)}+d_{j+1}^{(i)}}\right)^2 \eta_j^{j(i)} \\ &\leq \left(1-\frac{d_{j+1}^{(i)}}{s_j^{j(i)}+d_{j+1}^{(i)}}\right)^2 \kappa^2 {s_{j+1}^k}^{(i)} + \left(\frac{d_{j+1}^{(i)}}{s_j^{j(i)}+d_{j+1}^{(i)}}-1\right)^2 \kappa^2 d_{j+1}^{(i)} \\ &\quad + \left(\frac{d_{j+1}^{(i)}}{s_j^{j(i)}+d_{j+1}^{(i)}}\right)^2 \kappa^2 s_j^{k(i)}\end{aligned}$$

Using (1.10) we have that the sub-Gaussian parameter satisfies

$$\begin{aligned}\left(\eta_j^{k(i)}\right)^2 &= \left(1-\eta_j^{k(i)}\right)^2 \kappa^2 {s_{j+1}^k}^{(i)} + \left(\frac{-s_{j+1}^{(i)}}{s_j^{j(i)}+d_{j+1}^{(i)}}\right)^2 \kappa^2 d_{j+1}^{(i)} \\ &\quad + \left(\frac{d_{j+1}^{(i)}}{s_j^{j(i)}+d_{j+1}^{(i)}}\right)^2 \kappa^2 s_j^{k(i)} \\ &= \kappa^2 {s_{j+1}^k}^{(i)} + \kappa^2 \eta_j^{k(i)}\left(-2+\eta_j^{k(i)}\right) {s_{j+1}^k}^{(i)} + \eta_j^{k(i)}\left(1-\eta_j^{k(i)}\right)\kappa^2 s_j^{j(i)} \\ &\quad + \eta_j^{k(i)}\left(1-\eta_j^{k(i)}\right)\kappa^2 d_{j+1}^{(i)} \\ &= \kappa^2 {s_{j+1}^k}^{(i)} + \kappa^2 \eta_j^{k(i)}\left(\left(-2+\eta_j^{k(i)}\right) {s_{j+1}^k}^{(i)} + s_j^{j(i)}\right) = \kappa^2 s_j^{k(i)}\end{aligned}$$

The inequality in (A.18) is obtained using the union bound together with the Chernoff bound (concentration inequality) [85] for the random variables $z_j^{k(i)}$ that are sub-Gaussian with parameter $\eta_j^{k(i)}$.

Now, we prove by induction that, for any j, $\|\sqrt{\boldsymbol{s}_j^k}\| \leq M/\sqrt{n_j^k}$ where the ESSs satisfy $n_j^k \geq \frac{\left(\|\boldsymbol{\sigma}_j^2\|_\infty + n_j^j \|\boldsymbol{d}_{j+1}\|_\infty\right)^2 n_{j+1}^k}{\|\boldsymbol{\sigma}_j^2\|_\infty^2 + \|\boldsymbol{d}_{j+1}\|_\infty (\|\boldsymbol{\sigma}_j^2\|_\infty + n_j^j \|\boldsymbol{d}_{j+1}\|_\infty) n_{j+1}^k}$. Firstly, for $j = k$, using the proof of Theorem 2 in Appendix A.2, we have that $\|\sqrt{\boldsymbol{s}_k^k}\| \leq M/\sqrt{n_k^k}$. If $\|\sqrt{\boldsymbol{s}_{j+1}^k}\| \leq M/\sqrt{n_{j+1}^k}$,

then we that for every component i

$$\begin{aligned} s_j^{k(i)} &= s_{j+1}^{k\ (i)} + \eta_j^{k(i)}(s_j^{j(i)} - 2s_{j+1}^{k\ (i)} + \eta_j^{k(i)} s_{j+1}^{k\ (i)}) \\ &= \left(1 - \eta_j^{k(i)}\right)^2 s_{j+1}^{k\ (i)} + \eta_j^{k(i)} s_j^{j(i)} \\ &= \left(\frac{s_j^{j(i)}}{s_j^{j(i)} + d_{j+1}^{(i)}}\right)^2 s_{j+1}^{k\ (i)} + \frac{d_{j+1}^{(i)}}{s_j^{j(i)} + d_{j+1}^{(i)}} s_j^{j(i)} \end{aligned} \tag{A.20}$$

where the first inequality is obtained using the recursion for $\boldsymbol{s}_j^k$ in (1.9) and (A.20) is obtained using the gain vector in (1.10). Using that (A.20) is monotonically increasing for $s_j^{j(i)}$, we have that

$$\begin{aligned} s_j^{k(i)} &\leq \left(\frac{\frac{M^2}{n_j^j}}{\frac{M^2}{n_j^j} + d_{j+1}^{(i)}}\right)^2 \frac{M^2}{n_{j+1}^k} + \frac{d_{j+1}^{(i)}}{\frac{M^2}{n_j^j} + d_{j+1}^{(i)}} \frac{M^2}{n_j^j} \\ &= \left(\frac{M^2}{M^2 + n_j^j d_{j+1}^{(i)}}\right)^2 \frac{M^2}{n_{j+1}^k} + \frac{d_{j+1}^{(i)}}{M^2 + n_j^j d_{j+1}^{(i)}} M^2 \\ &= \frac{M^2}{M^2 + n_j^j d_{j+1}^{(i)}} \left(\frac{M^4}{M^2 + n_j^j d_{j+1}^{(i)}} \frac{1}{n_{j+1}^k} + d_{j+1}^{(i)}\right) \\ &= \frac{M^2}{M^2 + n_j^j d_{j+1}^{(i)}} \left(\frac{M^4 + d_{j+1}^{(i)}(M^2 + n_j^j d_{j+1}^{(i)}) n_{j+1}^k}{(M^2 + n_j^j d_{j+1}^{(i)}) n_{j+1}^k}\right) \end{aligned}$$

by using the induction hypothesis and $\|\sqrt{\boldsymbol{s}_j^j}\| \leq M/\sqrt{n_j^j}$ (proof of Theorem 2 in Appendix A.2). Then, we obtain

$$\|\sqrt{\boldsymbol{s}_j^k}\|_\infty \leq \frac{M}{\sqrt{\frac{\left(\|\boldsymbol{\sigma}_j^2\|_\infty + n_j^j \|\boldsymbol{d}_{j+1}\|_\infty\right)^2 n_{j+1}^k}{\|\boldsymbol{\sigma}_j^2\|_\infty^2 + \|\boldsymbol{d}_{j+1}\|_\infty (\|\boldsymbol{\sigma}_j^2\|_\infty + n_j^j \|\boldsymbol{d}_{j+1}\|_\infty) n_{j+1}^k}}}. \tag{A.21}$$

The inequality in (1.17) is obtained because the minimax risk is bounded by the smallest minimax risk as shown in [76–78] so that

$$R(\mathcal{U}_j^k) \leq R_j^\infty + \left(\|\boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_j^k\|_\infty + \|\boldsymbol{\lambda}_j^k\|_\infty\right) \left\|\boldsymbol{\mu}_j^\infty\right\|_1$$

that leads to (1.17) using (A.18), (A.21), and the fact that $1 \leq \sqrt{2\log\left(\frac{2m}{\delta}\right)}$. □

A.7 Proof of Theorem 5

Proof. The above recursions in (1.8)-(1.9) in Section 1.4 provide the same mean vector estimate as the recursions in (3.6)-(3.7) in Section 3.4.1 since they are obtained using

the Rauch-Tung-Striebel smoother recursions instead of fixed-lag smoother recursions. Hence, the ESS in (1.17) is the same as the ESS in (3.8). We obtain bound in (1.18) using the ESS in (3.8) (see Appendix C.2).

□

Appendix B

Minimax classification under concept drift

B.1 Proof of Theorem 6.

Proof. For the first bound in (2.3), we have that

$$R(\mathrm{h}_t) \leq \max_{\mathrm{p} \in \mathcal{U}_t^\infty} \ell(\mathrm{h}_t, \mathrm{p})$$

since $R(\mathrm{h}_t) = \ell(\mathrm{h}_t, \mathrm{p}_t)$ and $\mathrm{p}_t \in \mathcal{U}_t^\infty$ by definition of $\mathcal{U}_t^\infty$. Then, the maximization

$$\max_{\mathrm{p} \in \mathcal{U}_t^\infty} \ell(\mathrm{h}_t, \mathrm{p})$$

is obtained by using a slight reformulation of the optimization problem in the Theorem 2 in [76]. Specifically, with the notation in this chapter such optimization problem is

$$\begin{array}{ll} \min\limits_{\boldsymbol{\mu}, \nu} & -\boldsymbol{\tau}_t^{\mathrm{T}} \boldsymbol{\mu} - \nu \\ \text{s. t.} & \Phi(x,y)^{\mathrm{T}} \boldsymbol{\mu} + \nu \leq \mathrm{h}_t(y|x) - 1 \\ & \text{for any } (x,y) \in \mathcal{X} \times \mathcal{Y} \end{array}$$

In addition, the constraint of the above optimization problem can be rewritten as

$$\begin{gathered} \Phi(x,y)^{\mathrm{T}} \boldsymbol{\mu} + \nu \leq \mathrm{h}_t(y|x) - 1, \ \forall x, y \Leftrightarrow \nu \leq \mathrm{h}_t(y|x) - 1 - \Phi(x,y)^{\mathrm{T}} \boldsymbol{\mu}, \ \forall x, y \\ \Leftrightarrow \nu \leq - \max_{x \in \mathcal{X}, y \in \mathcal{Y}} \{\Phi(x,y)^{\mathrm{T}} \boldsymbol{\mu} - \mathrm{h}_t(y|x) + 1\}. \end{gathered}$$

Then, minimizing $-\nu$, we obtain the optimization problem

$$\max_{\mathrm{p} \in \mathcal{U}_t^\infty} \ell(\mathrm{h}_t, \mathrm{p}) = \min_{\boldsymbol{\mu}} 1 - \boldsymbol{\tau}_t^{\mathrm{T}} \boldsymbol{\mu} + \max_{x \in \mathcal{X}, y \in \mathcal{Y}} \left\{\Phi(x,y)^{\mathrm{T}} \boldsymbol{\mu} - \mathrm{h}_t(y|x)\right\}$$

that results in

$$\begin{aligned}\max_{\mathrm{p}\in\mathcal{U}_t^\infty} \ell(\mathrm{h}_t, \mathrm{p}) &= \min_{\boldsymbol{\mu}} 1 - \boldsymbol{\tau}_t^{\mathrm{T}}\boldsymbol{\mu} + \max_{x\in\mathcal{X},y\in\mathcal{Y}} \left\{\Phi(x,y)^{\mathrm{T}}\boldsymbol{\mu} - \mathrm{h}_t(y|x)\right\} \\ &\leq 1 - \boldsymbol{\tau}_t^{\mathrm{T}}\boldsymbol{\mu}_t + \max_{x\in\mathcal{X},y\in\mathcal{Y}} \left\{\Phi(x,y)^{\mathrm{T}}\boldsymbol{\mu}_t - \mathrm{h}_t(y|x)\right\} \\ &\leq 1 - \boldsymbol{\tau}_t^{\mathrm{T}}\boldsymbol{\mu}_t + \varphi(\boldsymbol{\mu}_t)\end{aligned}$$

where the last inequality is due to the definition of h_t and $\varphi(\cdot)$ in (4) and (5), respectively. Then, we have that

$$\begin{aligned}R(\mathrm{h}_t) &\leq 1 - \boldsymbol{\tau}_t^{\mathrm{T}}\boldsymbol{\mu}_t + \varphi(\boldsymbol{\mu}_t) \\ &= 1 - \boldsymbol{\tau}_t^{\mathrm{T}}\boldsymbol{\mu}_t + \varphi(\boldsymbol{\mu}_t) + \widehat{\boldsymbol{\tau}}_t^{\mathrm{T}}\boldsymbol{\mu}_t - \widehat{\boldsymbol{\tau}}_t^{\mathrm{T}}\boldsymbol{\mu}_t + \boldsymbol{\lambda}_t^{\mathrm{T}}|\boldsymbol{\mu}_t| - \boldsymbol{\lambda}_t^{\mathrm{T}}|\boldsymbol{\mu}_t| \\ &= R(\mathcal{U}_t) - \boldsymbol{\tau}_t^{\mathrm{T}}\boldsymbol{\mu}_t + \widehat{\boldsymbol{\tau}}_t^{\mathrm{T}}\boldsymbol{\mu}_t - \boldsymbol{\lambda}_t^{\mathrm{T}}|\boldsymbol{\mu}_t| \qquad \text{(B.1)}\end{aligned}$$

that leads to the first bound in (2.3) for $\alpha_t = \left\|\left|\boldsymbol{\tau}_t - \widehat{\boldsymbol{\tau}}_t\right| - \boldsymbol{\lambda}_t\right\|_\infty \left\|\boldsymbol{\mu}_t\right\|_1$ using Hölder's inequality. In addition, the minimax risk given by the optimization problem (6) satisfies

$$\begin{aligned}R(\mathcal{U}_t) &= \min_{\boldsymbol{\mu}} 1 - \widehat{\boldsymbol{\tau}}_t^{\mathrm{T}}\boldsymbol{\mu} + \varphi(\boldsymbol{\mu}) + \boldsymbol{\lambda}_t^{\mathrm{T}}\left|\boldsymbol{\mu}\right| \leq 1 - \widehat{\boldsymbol{\tau}}_t^{\mathrm{T}}\boldsymbol{\mu}_t^\infty + \varphi(\boldsymbol{\mu}_t^\infty) + \boldsymbol{\lambda}_t^{\mathrm{T}}\left|\boldsymbol{\mu}_t^\infty\right| \\ &= R_t^\infty + (\boldsymbol{\tau}_t - \widehat{\boldsymbol{\tau}}_t)^{\mathrm{T}}\boldsymbol{\mu}_t^\infty + \boldsymbol{\lambda}_t^{\mathrm{T}}\left|\boldsymbol{\mu}_t^\infty\right| \qquad \text{(B.2)}\end{aligned}$$

that together with (B.1) leads to the second bound in (2.3) for $\beta_t = (\left\|\boldsymbol{\tau}_t - \widehat{\boldsymbol{\tau}}_t\right\|_\infty + \left\|\boldsymbol{\lambda}_t\right\|_\infty)\left\|\boldsymbol{\mu}_t^\infty - \boldsymbol{\mu}_t\right\|$ since

$$\begin{aligned}R(\mathrm{h}_t) &\leq R(\mathcal{U}_t) - \boldsymbol{\tau}_t^{\mathrm{T}}\boldsymbol{\mu}_t + \widehat{\boldsymbol{\tau}}_t^{\mathrm{T}}\boldsymbol{\mu}_t - \boldsymbol{\lambda}_t^{\mathrm{T}}|\boldsymbol{\mu}_t| \\ &\leq R_t^\infty + (\boldsymbol{\tau}_t - \widehat{\boldsymbol{\tau}}_t)^{\mathrm{T}}(\boldsymbol{\mu}_t^\infty - \boldsymbol{\mu}_t) + \boldsymbol{\lambda}_t^{\mathrm{T}}(|\boldsymbol{\mu}_t^\infty| - |\boldsymbol{\mu}_t|) \\ &\leq R_t^\infty + (\boldsymbol{\tau}_t - \widehat{\boldsymbol{\tau}}_t)^{\mathrm{T}}(\boldsymbol{\mu}_t^\infty - \boldsymbol{\mu}_t) + \boldsymbol{\lambda}_t^{\mathrm{T}}\left|\boldsymbol{\mu}_t^\infty - \boldsymbol{\mu}_t\right| \qquad \text{(B.3)} \\ &\leq R_t^\infty + (\left\|\boldsymbol{\tau}_t - \widehat{\boldsymbol{\tau}}_t\right\|_\infty + \left\|\boldsymbol{\lambda}_t\right\|_\infty)\left\|\boldsymbol{\mu}_t^\infty - \boldsymbol{\mu}_t\right\|_1\end{aligned}$$

where (B.3) is obtained using the reverse triangle inequality since $\boldsymbol{\lambda}_t$ is positive.

For the first bound in (2.3) in the case that $\boldsymbol{\lambda}_t \succeq \left|\boldsymbol{\tau}_t - \widehat{\boldsymbol{\tau}}_t\right|$, we have that $R(\mathrm{h}_t) \leq R(\mathcal{U}_t)$ because $\mathrm{p}_t \in \mathcal{U}_t$ by definition of $\mathcal{U}_t$. Then, the minimax risk given by the optimization problem (6) satisfies (B.2) that leads to (2.3) for $\alpha_t = 0$ and $\beta_t = 2\left\|\boldsymbol{\lambda}_t\right\|_\infty \left\|\boldsymbol{\mu}_t^\infty\right\|_1$ since $\boldsymbol{\lambda}_t \succeq \left|\boldsymbol{\tau}_t - \widehat{\boldsymbol{\tau}}_t\right|$.

For the result in (2.4), let $(x_1, y_1), (x_2, y_2), ..., (x_T, y_T)$ be a sequence of instance-label pairs. If

$$V_t \;=\; \mathbb{1}\left\{\hat{y}_t \neq y_t\right\} \;-\; R(\mathrm{h}_t)$$

we have that the sequence $V_1, V_2, ...$ is a martingale difference with respect to $(x_1, y_1), (x_2, y_2), ...$ because

$$\mathbb{E}[V_t|(x_1, y_1), (x_2, y_2), ..., (x_{t-1}, y_{t-1})] = 0$$

for any t. Then, using the Azuma's inequality [73], since $|V_t| \leq 1$ for any t, we have that

$$\sum_{t=1}^{T} V_t \leq \sqrt{2T\log\frac{1}{\delta}} \qquad \text{(B.4)}$$

with probability at least $1-\delta$. Then, the result is obtained using bound in (2.3) because with probability at least $1-\delta$ we have that

$$\sum_{t=1}^{T} \mathbb{1}\left\{\hat{y}_t \neq y_t\right\} \leq \sum_{t=1}^{T} R(\mathrm{h}_t) + \sqrt{2T \log \frac{1}{\delta}}$$

using (B.4) and the definition of V_t. □

B.2 Derivarion of the dynamical system

Dynamical systems as that given by (2.6) can be derived using the k-th order Taylor expansion of the conditional expectations $\gamma_{t,i} = \mathbb{E}_{\mathrm{p}_t}\{\Psi_r(x)|y=j\}$ with $i=(d-1)j+r$ for $j=1,2,...,|\mathcal{Y}|$ and $r=1,2,...,d$. The Taylor expansions of $\gamma_{t,i}$ and its successive derivatives up to order k are given by

$$\begin{aligned}
\gamma_{t,i} &\approx \gamma_{t-1,i} + \Delta_t \gamma'_{t-1,i} + \frac{\Delta_t^2}{2}\gamma''_{t-1,i} + ... + \frac{\Delta_t^k}{k!}\gamma^{k)}_{t-1,i} \\
\gamma'_{t,i} &\approx \gamma'_{t-1,i} + \Delta_t \gamma''_{t-1,i} + ... + \frac{\Delta_t^{k-1}}{(k-1)!}\gamma^{k)}_{t-1,i} \\
&\vdots \\
\gamma^{k)}_{t,i} &\approx \gamma^{k)}_{t-1,i}
\end{aligned}$$

where Δ_t is the time increment at t and $\gamma^{k)}_{t,i}$ denotes the k-th derivative of $\gamma_{t,i}$. The above equations lead to

$$\boldsymbol{\eta}_{t,i} \approx \begin{pmatrix} 1 & \Delta_t & \frac{\Delta_t^2}{2} & ... & \frac{\Delta_t^k}{k!} \\ 0 & 1 & \Delta_t & ... & \frac{\Delta_t^{k-1}}{(k-1)!} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & ... & 1 \end{pmatrix} \boldsymbol{\eta}_{t-1,i} \tag{B.5}$$

since $\boldsymbol{\eta}_{t,i}$ is composed by $\gamma_{t,i}$ and its derivatives up to order k. In addition, $\gamma_{t,i}$ is observed at each time t through the instance-label pair (x_t, y_t), so that we have

$$\Phi_i(x_t, y_t) \approx \gamma_{t,i}, \text{ if } y_t = j \tag{B.6}$$

with $i=(d-1)j+r$ for $j=1,2,...,|\mathcal{Y}|$ and $r=1,2,...,d$. Then, equations (B.5) and (B.6) above lead to the dynamical system in (2.6).

Appendix C

Appendix Continual Learning

C.1 Proof of Theorem 10

Proof. To obtain bound in (3.8) we first prove that the mean vector estimate and the MSE vector given by (3.6) and (3.7), respectively, satisfy

$$\mathbb{P}\left\{|\tau_k^{\infty(i)} - \tau_j^{\rightleftharpoons k(i)}| \leq \kappa\sqrt{2s_j^{\rightleftharpoons k(i)}\log\left(\frac{2m}{\delta}\right)}\right\} \geq (1-\delta) \tag{C.1}$$

for any component $i = 1, 2, \ldots, m$. Then, we prove that $\|\boldsymbol{s}_j^{\rightleftharpoons k}\|_\infty \leq M/\sqrt{n_j^{\rightleftharpoons k}}$ for $j \in \{1, 2, \ldots, k\}$, where the ESSs satisfy $n_k^{\rightleftharpoons k} = n_k^{\rightharpoonup}$ and $n_j^{\rightleftharpoons k} \geq n_j^{\rightharpoonup} + n_{j+1}^{\leftharpoondown k}\frac{\|\boldsymbol{\sigma}_j^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty + n_{j+1}^{\leftharpoondown k}\|\boldsymbol{d}_{j+1}^2\|_\infty}$ for $j \geq 2$.

To obtain inequality (C.1), we prove that each component $i = 1, 2, \ldots, m$ of the error in the mean vector estimate $z_j^{\rightleftharpoons k(i)} = \tau_j^{\infty(i)} - \tau_j^{\rightleftharpoons k(i)}$ is sub-Gaussian with parameter $\eta_j^{\rightleftharpoons k(i)} \leq \kappa\sqrt{s_j^{\rightleftharpoons k(i)}}$. Analogously to the proof of Theorem 8, it is proven that each component in the error of the backward mean vector $\boldsymbol{\tau}_{j+1}^{\leftharpoondown k}$ is sub-Gaussian with parameters satisfying $\boldsymbol{\eta}_{j+1}^{\leftharpoondown k} \preceq \kappa\sqrt{\boldsymbol{s}_{j+1}^{\leftharpoondown k}}$. The error in the forward and backward mean vector estimate is given by

$$\begin{aligned} z_j^{\rightleftharpoons k(i)} &= \tau_j^{\infty(i)} - \tau_j^{\rightleftharpoons k(i)} \\ &= \tau_j^{\infty(i)} - \tau_j^{\rightharpoonup(i)} - \frac{s_j^{\rightharpoonup(i)}}{s_j^{\rightharpoonup(i)} + s_{j+1}^{\leftharpoondown k\,(i)} + d_{j+1}^{2\,(i)}}\left(\tau_{j+1}^{\leftharpoondown k(i)} - \tau_j^{\rightharpoonup(i)}\right) \end{aligned}$$

where the second equality is obtained using the recursion for $\tau_j^{\rightleftharpoons k(i)}$ in (3.6). Adding

and subtracting $\frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)}+s_{j+1}^{\leftarrow k\,(i)}+d_{j+1}^{2\ (i)}}\tau_{j+1}^{\infty\ (i)}$, we have that

$$\begin{aligned} z_j^{\rightleftharpoons k(i)} &= z_j^{\rightarrow(i)} - \frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)} + s_{j+1}^{\leftarrow k\,(i)} + d_{j+1}^{2\ (i)}} \left(\tau_{j+1}^{\infty\ (i)} - \tau_{j+1}^{\infty\ (i)} + \tau_{j+1}^{\leftarrow k(i)} - \tau_j^{\rightarrow(i)}\right) \\ &= z_j^{\rightarrow(i)} - \frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)} + s_{j+1}^{\leftarrow k\,(i)} + d_{j+1}^{2\ (i)}} \left(\tau_j^{\infty(i)} + w_{j+1}^{(i)} - z_{j+1}^{\leftarrow k\,(i)} - \tau_j^{\rightarrow(i)}\right) \end{aligned}$$

since $\boldsymbol{w}_j = \boldsymbol{\tau}_j^{\infty} - \boldsymbol{\tau}_{j-1}^{\infty}$ and $z_j^{\rightarrow(i)} = \tau_j^{\infty(i)} - \tau_j^{\rightarrow(i)}$. Then, we have that

$$\begin{aligned} z_j^{\rightleftharpoons k(i)} =& z_j^{\rightarrow(i)} - \frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)} + s_{j+1}^{\leftarrow k\,(i)} + d_{j+1}^{2\ (i)}} \left(z_j^{\rightarrow(i)} + w_{j+1}^{(i)} - z_{j+1}^{\leftarrow k\,(i)}\right) \\ =& \left(1 - \frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)} + s_{j+1}^{\leftarrow k\,(i)} + d_{j+1}^{2\ (i)}}\right) z_j^{\rightarrow(i)} \\ & - \frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)} + s_{j+1}^{\leftarrow k\,(i)} + d_{j+1}^{2\ (i)}} \left(w_{j+1}^{(i)} - z_{j+1}^{\leftarrow k\,(i)}\right) \end{aligned} \tag{C.2}$$

where $z_j^{\rightarrow(i)}$, $z_{j+1}^{\leftarrow k\,(i)}$, and $w_{j+1}^{(i)}$ are sub-Gaussian with parameters $\eta_j^{\rightarrow(i)} \leq \kappa\sqrt{s_j^{\rightarrow(i)}}$, $\eta_{j+1}^{\leftarrow k\,(i)} \leq \kappa\sqrt{s_{j+1}^{\leftarrow k\,(i)}}$, and $\sigma(w_j^{(i)})$, respectively. Since $\boldsymbol{z}_j^{\rightarrow}$, $\boldsymbol{z}_{j+1}^{\leftarrow k}$, and $\boldsymbol{w}_{j+1}$ are independent, we have that $z_j^{\rightleftharpoons k(i)}$ given by (C.2) is sub-Gaussian with parameter that satisfies

$$\begin{aligned} \left(\eta_j^{\rightleftharpoons k(i)}\right)^2 =& \left(1 - \frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)} + s_{j+1}^{\leftarrow k\,(i)} + d_{j+1}^{2\ (i)}}\right)^2 \left(\eta_j^{\rightarrow(i)}\right)^2 \\ & + \left(\frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)} + s_{j+1}^{\leftarrow k\,(i)} + d_{j+1}^{2\ (i)}}\right)^2 \left(\sigma\left(w_j^{(i)}\right)^2 + \left(\eta_{j+1}^{\leftarrow k\,(i)}\right)^2\right) \\ \leq& \left(1 - \frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)} + s_{j+1}^{\leftarrow k\,(i)} + d_{j+1}^{2\ (i)}}\right)^2 \kappa^2 s_j^{\rightarrow(i)} \\ & + \left(\frac{s_j^{\rightarrow(i)}}{s_j^{\rightarrow(i)} + s_{j+1}^{\leftarrow k\,(i)} + d_{j+1}^{2\ (i)}}\right)^2 \kappa^2 \left(d_{j+1}^{\ (i)} + s_{j+1}^{\leftarrow k\,(i)}\right) \end{aligned}$$

Using (3.7) we have that the sub-Gaussian parameter satisfies

$$\begin{aligned}\left(\eta_j^{\rightleftharpoons k(i)}\right)^2 \leq & \left(1-\frac{s_j^{\rightleftharpoons k(i)}}{s_{j+1}^{\leftarrow k\,(i)}+{d_{j+1}^2}^{(i)}}\right)^2 \kappa^2 \left(\frac{1}{s_j^{\rightleftharpoons k(i)}}-\frac{1}{s_{j+1}^{\leftarrow k^2\,(i)}+{d_{j+1}^2}^{(i)}}\right)^{-1} \\ & + \frac{\left(s_j^{\rightleftharpoons k(i)}\right)^2}{s_{j+1}^{\leftarrow k\,(i)}+{d_{j+1}^2}^{(i)}}\kappa^2 \\ = & \left(\frac{s_{j+1}^{\leftarrow k\,(i)}+{d_{j+1}^2}^{(i)}-s_j^{\rightleftharpoons k(i)}}{s_{j+1}^{\leftarrow k\,(i)}+{d_{j+1}^2}^{(i)}}\right)\kappa^2 s_j^{\rightleftharpoons k(i)} + \frac{\left(s_j^{\rightleftharpoons k(i)}\right)^2}{s_{j+1}^{\leftarrow k\,(i)}+{d_{j+1}^2}^{(i)}}\kappa^2 \\ = & \kappa^2 s_j^{\rightleftharpoons k(i)}.\end{aligned}$$

The inequality in (C.1) is obtained using the union bound together with the Chernoff bound (concentration inequality) [85] for the random variables $z_j^{\rightleftharpoons k(i)}$ that are sub-Gaussian with parameter $\eta_j^{\rightleftharpoons k(i)}$.

Now, we prove that, for any j, $\|\sqrt{\boldsymbol{s}_j^{\rightleftharpoons k}}\| \leq M/\sqrt{n_j^{\rightleftharpoons k}}$ where the ESSs satisfy $n_k^{\rightleftharpoons k} = n_k^{\rightarrow}$ and $n_j^{\rightleftharpoons k} \geq n_j^{\rightarrow} + n_{j+1}^{\leftarrow k}\frac{\|\boldsymbol{\sigma}_j^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty + n_{j+1}^{\leftarrow k}\|\boldsymbol{d}_{j+1}^2\|_\infty}$ for $j \geq 2$. Analogously to the proof of Theorem 8, we prove that the backward MSE vector $\boldsymbol{s}_{j+1}^{\leftarrow k}$ satisfies $\|\sqrt{\boldsymbol{s}_{j+1}^{\leftarrow k}}\|_\infty \leq M/\sqrt{n_{j+1}^{\leftarrow k}}$. Then, using that $\|\sqrt{\boldsymbol{s}_{j+1}^{\leftarrow k}}\|_\infty \leq M/\sqrt{n_{j+1}^{\leftarrow k}}$, we have that for every component i

$$\begin{aligned}\left(s_j^{\rightleftharpoons k(i)}\right)^{-1} &= \frac{1}{s_j^{\rightarrow(i)}} + \frac{1}{s_{j+1}^{\leftarrow k\,(i)}+{d_{j+1}^2}^{(i)}} \geq \frac{n_j^{\rightarrow}}{\sigma_j^{2(i)}} + \frac{1}{\frac{M^2}{n_{j+1}^{\leftarrow k}}+{d_{j+1}^2}^{(i)}} \\ &\geq \frac{1}{M^2}\left(n_j^{\rightarrow} + \frac{1}{\frac{1}{n_{j+1}^{\leftarrow k}}+\frac{d_{j+1}^2}{M^2}}\right) \geq \frac{1}{M^2}\left(n_j^{\rightarrow} + \frac{1}{\frac{1}{n_{j+1}^{\leftarrow k}}+\frac{\|d_{j+1}^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty}}\right).\end{aligned}$$

Then, we obtain

$$\|\sqrt{\boldsymbol{s}_j^{\rightleftharpoons k}}\|_\infty \leq \frac{M}{\sqrt{n_j^{\rightarrow} + \frac{1}{\frac{1}{n_{j+1}^{\leftarrow k}}+\frac{\|\boldsymbol{d}_{j+1}^2\|_\infty}{\|\boldsymbol{\sigma}_j^2\|_\infty}}}}. \tag{C.3}$$

The inequality in (3.8) is obtained because the minimax risk is bounded by the smallest minimax risk as shown in [76–78] so that

$$R(\mathcal{U}_j^{\rightleftharpoons k}) \leq R_j^\infty + \left(\|\boldsymbol{\tau}_j^\infty - \boldsymbol{\tau}_j^{\rightleftharpoons k}\|_\infty + \|\boldsymbol{\lambda}_j^{\rightleftharpoons k}\|_\infty\right)\left\|\boldsymbol{\mu}_j^\infty\right\|_1$$

that leads to (3.8) using (C.1), (C.3), and the fact that $1 \leq \sqrt{2\log\left(\frac{2m}{\delta}\right)}$. □

C.2 Proof of Theorem 11

Proof. To obtain bound in (3.9), we use the ESS obtained with forward learning in Theorem 9 and obtained with backward learning. Analogously to the proof of Theorem 9, we prove that the ESS obtained at backward learning satisfies

$$\begin{aligned} n_{j+1}^{\leftarrow k} &\geq n_{j+1} + n_{j+2}^{\leftarrow k} \frac{\|\boldsymbol{\sigma}_{j+1}^2\|_\infty}{\|\boldsymbol{\sigma}_{j+1}^2\|_\infty + n_{j+2}^{\leftarrow k}\|\boldsymbol{d}_{j+2}^2\|_\infty} \\ &\geq n\left(1 + \frac{(1+\alpha)^{2(k-j)-1} - 1 - \alpha}{\alpha(1+\alpha)^{2(k-j)-1} + \alpha}\right). \end{aligned}$$

Therefore, the ESS obtained with forward an backward learning satisfies

$$\begin{aligned} n_j^{\rightleftharpoons k} \geq & n_j^{\rightarrow} + n\left(1 + \frac{(1+\alpha)^{2(k-j)-1} - 1 - \alpha}{\alpha(1+\alpha)^{2(k-j)-1} + \alpha}\right) \\ & \cdot \left(1 + \frac{n\left(1 + \frac{(1+\alpha)^{2(k-j)-1} - 1 - \alpha}{\alpha(1+\alpha)^{2(k-j)-1} + \alpha}\right)}{nd^2}\right)^{-1} \\ = & n_j^{\rightarrow} + n\frac{(1+\alpha)^{2(k-j)} - 1}{\alpha(1+\alpha)^{2(k-j)-1} + \alpha} \\ & \cdot \left(1 + \frac{\alpha^2}{\alpha+1}\left(1 + \frac{(1+\alpha)^{2(k-j)-1} - 1 - \alpha}{\alpha(1+\alpha)^{2(k-j)-1} + \alpha}\right)\right)^{-1} \end{aligned}$$

where the second equality follows because $nd^2 = \frac{\alpha^2}{\alpha+1}$ since

$$\alpha = \frac{nd^2}{2}\left(\sqrt{1 + \frac{4}{nd^2}} + 1\right).$$

Then, we have that

$$\begin{aligned} n_j^{\rightleftharpoons k} \geq & n_j^{\rightarrow} + n\frac{(1+\alpha)^{2(k-j)} - 1}{\alpha(1+\alpha)^{2(k-j)-1} + \alpha} \\ & \cdot \left(\frac{((1+\alpha)^{2(k-j)-1} + 1)(\alpha + 1 + \alpha^2) + \alpha((1+\alpha)^{2(k-j)-1} - 1 - \alpha)}{(\alpha+1)((1+\alpha)^{2(k-j)-1} + 1)}\right)^{-1} \\ \geq & n_j^{\rightarrow} + n\frac{(1+\alpha)^{2(k-j)} - 1}{\alpha(1+\alpha)^{2(k-j)-1} + \alpha}\frac{(\alpha+1)((1+\alpha)^{2(k-j)-1} + 1)}{(1+\alpha)^{2(k-j)+1} + 1}. \end{aligned}$$

Now, we obtain bounds for the ESS depending on the value value of nd^2. Such bounds are obtained similarly as in Theorem 9 and we also denote by ϕ the golden ratio $\phi = 1.618\ldots$.

1. If $nd^2 < \frac{1}{j^2} \Rightarrow \sqrt{nd^2} \leq \alpha \leq \sqrt{nd^2}\phi \leq \frac{\phi}{j} \leq 1$ similarly as in the previous case, then we have that $n_j^{\rightleftharpoons k}$ satisfies

$$\begin{aligned} n_j^{\rightleftharpoons k} &\geq n_j^{\rightharpoonup} + n\frac{1}{\alpha}\frac{\alpha(2(k-j))}{2+\alpha 2(k-j)} = n_j^{\rightharpoonup} + n\frac{k-j}{1+\alpha(k-j)} \\ &\geq n_j^{\rightharpoonup} + n\frac{k-j}{1+\frac{\phi}{j}(k-j)} \end{aligned}$$

   where the first inequality follows because

$$(1+\alpha)^{2(k-j)-1} \geq 1 + \alpha(2(k-j)-1)$$

   and the second inequality is obtained using $\alpha \leq \frac{\phi}{j}$.

2. If $\frac{1}{j^2} \leq nd^2 < 1 \Rightarrow \frac{1}{j} \leq \sqrt{nd^2} \leq \alpha \leq \sqrt{nd^2}\phi$ because

$$\alpha = nd^2\frac{\sqrt{1+\frac{4}{nd^2}}+1}{2} = \sqrt{nd^2}\frac{\sqrt{nd^2+4}+\sqrt{nd^2}}{2}$$

   then we have that $n_j^{\rightleftharpoons k}$ satisfies

$$n_j^{\rightleftharpoons k} \geq n_j^{\rightharpoonup}\frac{n}{\alpha}\frac{(1+\alpha)^{2(k-j)}-1}{(1+\alpha)^{2(k-j)}+1} \geq n_j^{\rightharpoonup}\frac{n}{\alpha}\frac{(1+\sqrt{nd^2})^{2(k-j)}-1}{(1+\sqrt{nd^2})^{2(k-j)}+1}$$

   where the second inequality follows because the ESS is monotonically increasing for α and $\alpha \geq nd^2$. Since $(1+\sqrt{nd^2})^{2(k-j)} \geq 1 + 2\sqrt{nd^2}(k-j)$ and $k-j \geq 1$, we have that

$$n_j^{\rightleftharpoons k} \geq n_j^{\rightharpoonup} + \frac{n}{\alpha}\frac{\sqrt{nd^2}}{1+\sqrt{nd^2}} \geq n_j^{\rightharpoonup} + n\frac{1}{\phi}\frac{1}{1+\sqrt{nd^2}}$$

   because $\alpha \leq \sqrt{nd^2}\phi$.

3. If $nd^2 \geq 1 \Rightarrow 1 \leq nd^2 \leq \alpha \leq nd^2\phi$ because $\alpha = nd^2\frac{\sqrt{1+\frac{4}{nd^2}}+1}{2}$, then we have that $n_j^{\rightleftharpoons k}$ satisfies

$$n_j^{\rightleftharpoons k} \geq n_j^{\rightharpoonup} + n\frac{1}{\alpha}\frac{2^{2(k-j)}-1}{2^{2(k-j)}+1} \geq n_j^{\rightharpoonup} + n\frac{1}{nd^2}\frac{1}{\phi}\frac{3}{5}$$

   where the first inequality follows because the ESS is monotonically increasing for α and $\alpha \geq 1$ and the second inequality is obtained using $k-j \geq 1$ and $\alpha \leq nd^2\phi$.

□

Appendix D

Probabilistic Load Forecasting

D.1 Proof of Theorem 12

Proof. We first prove that for any $i > 0$ the optimal parameters $\boldsymbol{\eta}_i^*$, σ_i^* satisfy

$$\sum_{j=1}^{i} \lambda^{i-j} \mathbf{u}_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \boldsymbol{\eta}_i^* = \mathbf{q}_i \tag{D.1}$$

$$\gamma_i {\sigma_i^*}^2 = \sum_{j=1}^{i} \lambda^{i-j} s_{t_j}^2 - \mathbf{q}_i^{\mathrm{T}} \boldsymbol{\eta}_i^* \tag{D.2}$$

while parameters $\boldsymbol{\eta}_i$, σ_i and matrix $\mathbf{P}_i$ given by recursions (4.7)-(4.10) with $\boldsymbol{\eta}_0 = \mathbf{0}_K$, any σ_0, $\mathbf{P}_0 = \mathbf{I}_K$, and $\gamma_0 = 0$ satisfy

$$\mathbf{P}_i^{-1} \boldsymbol{\eta}_i = \mathbf{q}_i \tag{D.3}$$

$$\gamma_i {\sigma_i}^2 = \sum_{j=1}^{i} \lambda^{i-j} s_{t_j}^2 - \mathbf{q}_i^{\mathrm{T}} \boldsymbol{\eta}_i \tag{D.4}$$

$$\mathbf{P}_i^{-1} = \lambda^i \mathbf{I}_K + \sum_{j=1}^{i} \lambda^{i-j} \mathbf{u}_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \tag{D.5}$$

where $\mathbf{q}_i = \sum_{j=1}^{i} \lambda^{i-j} s_{t_j} \mathbf{u}_{t_j}$. Then, in the second step of the proof we obtain bound in (4.11) using equations (D.1)-(D.5). Finally, we prove that parameters $\boldsymbol{\eta}_i$, and σ_i given by recursions (4.7)-(4.10) with $\boldsymbol{\eta}_{i_0}$, σ_{i_0} given by (4.12)-(4.15) satisfy $\boldsymbol{\eta}_i = \boldsymbol{\eta}_i^*$ and $\sigma_i = \sigma_i^*$, for $i \geq i_0$.

Parameters $\boldsymbol{\eta}_i^*$ and σ_i^* satisfy equations (D.1) and (D.2), respectively, because they maximize the log-likelihood in (4.5). The differentiable function $L_i(\boldsymbol{\eta}, \sigma)$ is concave since Gaussian functions are log-concave. Then, $L_i(\boldsymbol{\eta}, \sigma)$ has a maximum achieved by

parameters that result in zero derivatives. Since

$$L_i\left(\boldsymbol{\eta},\sigma\right)=-\sum_{j=1}^{i}\lambda^{i-j}\frac{\left(s_{t_j}-\mathbf{u}_{t_j}^{\mathrm{T}}\boldsymbol{\eta}\right)^2}{2\sigma^2}+\lambda^{i-j}\log\left(\sigma\sqrt{2\pi}\right)$$

we have that

$$\frac{\partial L_i(\boldsymbol{\eta},\sigma)}{\partial\boldsymbol{\eta}}=\sum_{j=1}^{i}\lambda^{i-j}\frac{\mathbf{u}_{t_j}(s_{t_j}-\mathbf{u}_{t_j}^{\mathrm{T}}\boldsymbol{\eta})}{\sigma^2}$$

that becomes zero for $\boldsymbol{\eta}_i^*$ given by (D.1), and

$$\frac{\partial L_i(\boldsymbol{\eta},\sigma)}{\partial\sigma}=\sum_{j=1}^{i}\lambda^{i-j}\frac{\left(s_{t_j}-\mathbf{u}_{t_j}^{\mathrm{T}}\boldsymbol{\eta}\right)^2}{\sigma^3}-\lambda^{i-j}\frac{1}{\sigma}$$

that becomes zero for σ_i^* given by (D.2) since γ_i given by (4.10) equals $\gamma_i=\sum_{j=1}^{i}\lambda^{i-j}$.

By induction, we prove that $\boldsymbol{\eta}_i$ and σ_i given by recursions (4.7) and (4.8) satisfy equations (D.3) and (D.4) for $\boldsymbol{\eta}_0=\mathbf{0}_K$, any σ_0, $\mathbf{P}_0=\mathbf{I}_K$, and $\gamma_0=0$. Firstly, we prove it for $i=1$. From (4.9), we have that

$$\mathbf{P}_1=\frac{1}{\lambda}\left(\mathbf{I}_K-\frac{\mathbf{u}_{t_1}\mathbf{u}_{t_1}^{\mathrm{T}}}{\lambda+\mathbf{u}_{t_1}^{\mathrm{T}}\mathbf{u}_{t_1}}\right)=\left(\lambda\mathbf{I}_K+\mathbf{u}_{t_1}\mathbf{u}_{t_1}^{\mathrm{T}}\right)^{-1}\tag{D.6}$$

applying the matrix inversion Lemma and using that $\mathbf{P}_0=\mathbf{I}_K$. Hence, from (4.7), (4.8), and (4.10), we get

$$\begin{aligned}\mathbf{P}_1^{-1}\boldsymbol{\eta}_1&=\left(\lambda\mathbf{I}_K+\mathbf{u}_{t_1}\mathbf{u}_{t_1}^{\mathrm{T}}\right)\frac{s_{t_1}\mathbf{u}_{t_1}}{\lambda+\mathbf{u}_{t_1}^{\mathrm{T}}\mathbf{u}_{t_1}}=s_{t_1}\mathbf{u}_{t_1}\\ \sigma_1^2&=\frac{\lambda s_{t_1}^2}{\lambda+\mathbf{u}_{t_1}^{\mathrm{T}}\mathbf{u}_{t_1}}=\frac{\lambda s_{t_1}^2+s_{t_1}^2\mathbf{u}_{t_1}^{\mathrm{T}}\mathbf{u}_{t_1}-s_{t_1}^2\mathbf{u}_{t_1}^{\mathrm{T}}\mathbf{u}_{t_1}}{\lambda+\mathbf{u}_{t_1}^{\mathrm{T}}\mathbf{u}_{t_1}}\\ &=s_{t_1}^2-s_{t_1}\mathbf{u}_{t_1}^{\mathrm{T}}\frac{s_{t_1}\mathbf{u}_{t_1}}{\lambda+\mathbf{u}_{t_1}^{\mathrm{T}}\mathbf{u}_{t_1}}=s_{t_1}^2-s_{t_1}\mathbf{u}_{t_1}^{\mathrm{T}}\boldsymbol{\eta}_1\end{aligned}$$

since $\boldsymbol{\eta}_0=\mathbf{0}_K$, $\mathbf{P}_0=\mathbf{I}_K$, and $\gamma_0=0$.

If (D.3) and (D.4) hold for $i-1$, then for i we have that

$$\mathbf{P}_i=\left(\lambda{\mathbf{P}_{i-1}}^{-1}+\mathbf{u}_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\right)^{-1}\tag{D.7}$$

applying the matrix inversion Lemma to equation (4.9). Therefore, using the recursion

of $\boldsymbol{\eta}_i$ in (4.7), we have that

$$\begin{aligned}
\mathbf{P}_i^{-1}\boldsymbol{\eta}_i =& \lambda \mathbf{P}_{i-1}{}^{-1}\boldsymbol{\eta}_{i-1} + \frac{\lambda \mathbf{u}_{t_i}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right) \\
&+ \mathbf{u}_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1} + \frac{\mathbf{u}_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right) \\
=& \lambda \mathbf{q}_{i-1} + \frac{\lambda \mathbf{u}_{t_i}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right) \\
&+ \mathbf{u}_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1} + \frac{\mathbf{u}_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right) \qquad \text{(D.8)} \\
=& \lambda \mathbf{q}_{i-1} + \mathbf{u}_{t_i}\left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right) + \mathbf{u}_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1} \\
=& \sum_{j=1}^{i-1}\lambda^{i-j}s_{t_j}\mathbf{u}_{t_j} + s_{t_i}\mathbf{u}_{t_i} = \mathbf{q}_i
\end{aligned}$$

where the equality (D.8) is obtained by using the induction hypothesis. Using the recursion of σ_i in (4.8), we have that

$$\begin{aligned}
\gamma_i {\sigma_i}^2 =& \left(\gamma_i - 1\right){\sigma_{i-1}}^2 \\
&+ \left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right)\left(s_{t_i} - \frac{s_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}} - \frac{\lambda \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\right) \\
=& \sum_{j=1}^{i-1}\lambda^{i-j}s_{t_j}^2 - \lambda \mathbf{q}_{i-1}^{\mathrm{T}}\boldsymbol{\eta}_{i-1} + \left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right) \qquad \text{(D.9)} \\
&\cdot \left(s_{t_i} - \frac{s_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}} - \frac{\lambda \mathbf{q}_{i-1}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\right) \qquad \text{(D.10)} \\
=& \sum_{j=1}^{i-1}\lambda^{i-j}s_{t_j}^2 + s_{t_i}^2 - \left(\sum_{j=1}^{i-1}\lambda^{i-j}s_{t_j}\mathbf{u}_{t_j}^{\mathrm{T}} + s_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\right) \\
&\cdot \left(\boldsymbol{\eta}_{i-1} + \frac{\mathbf{P}_{i-1}\mathbf{u}_{t_i}}{\lambda + \mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\left(s_{t_i} - \mathbf{u}_{t_i}^{\mathrm{T}}\boldsymbol{\eta}_{i-1}\right)\right) \qquad \text{(D.11)}
\end{aligned}$$

where the equality (D.10) is obtained by using the induction hypothesis. Then, we obtain (D.4) from (D.11) by using the recursion for $\boldsymbol{\eta}_i$ in (4.7).

In addition, by induction we prove that $\mathbf{P}_i$ given by recursion (4.9) satisfies the equation (D.5). The case $i = 1$ is proved in the equation (D.6). If (D.5) holds for $i-1$, then for i by using (D.7) and the induction hypothesis we have that

$$\mathbf{P}_i = \left(\lambda\left(\lambda^{i-1}\mathbf{I}_K + \sum_{j=1}^{i-1}\lambda^{i-1-j}\mathbf{u}_{t_j}\mathbf{u}_{t_j}^{\mathrm{T}}\right) + \mathbf{u}_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}}\right)^{-1}$$

that directly leads to (D.5).

To obtain the bound in (4.11), we first use the definition of L_i and equations (D.2) and (D.4) to obtain

$$\begin{aligned} L_i\left(\boldsymbol{\eta}_i^*, \sigma_i^*\right) - L_i\left(\boldsymbol{\eta}_i, \sigma_i\right) &= \frac{\gamma_i}{2} \log\left(\frac{\sigma_i^2}{\sigma_i^{*2}}\right) = \frac{\gamma_i}{2} \log\left(1 + \frac{\sigma_i^2 - \sigma_i^{*2}}{\sigma_i^{*2}}\right) \\ &\leq \frac{\gamma_i}{2} \frac{\left|\sigma_i^2 - \sigma_i^{*2}\right|}{\sigma_i^{*2}} = \frac{1}{2} \left| \frac{\mathbf{q}_i^{\mathrm{T}}\left(\boldsymbol{\eta}_i^* - \boldsymbol{\eta}_i\right)}{\sigma_i^{*2}} \right|. \end{aligned} \tag{D.12}$$

We then use the following inequalities

$$\left|\mathbf{q}_i^{\mathrm{T}}\left(\boldsymbol{\eta}_i^* - \boldsymbol{\eta}_i\right)\right| \leq \lambda^i \left\|\boldsymbol{\eta}_i^*\right\| \left\|\boldsymbol{\eta}_i\right\| \tag{D.13}$$

$$\frac{1}{\sigma_i^{*2}} \leq 2\pi M^2 \tag{D.14}$$

where the inequality (D.13) is obtained using equations (D.1) and (D.3) because

$$\begin{aligned} \left|\mathbf{q}_i^{\mathrm{T}}\left(\boldsymbol{\eta}_i^* - \boldsymbol{\eta}_i\right)\right| &= \left| \mathbf{q}_i^{\mathrm{T}} \left(\left(\sum_{j=1}^{i} \lambda^{i-j} \mathbf{u}_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \right)^{-1} - \mathbf{P}_i \right) \mathbf{q}_i \right| \\ &= \left| \mathbf{q}_i^{\mathrm{T}} \left(\sum_{j=1}^{i} \lambda^{i-j} \mathbf{u}_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \right)^{-1} \left(\mathbf{P}_i^{-1} - \sum_{j=1}^{i} \lambda^{i-j} \mathbf{u}_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \right) \mathbf{P}_i \mathbf{q}_i \right| \\ &= \left| \lambda^i \mathbf{q}_i^{\mathrm{T}} \left(\sum_{j=1}^{i} \lambda^{i-j} \mathbf{u}_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \right)^{-1} \mathbf{P}_i \mathbf{q}_i \right| = \lambda^i \left| \boldsymbol{\eta}_i^{*\mathrm{T}} \boldsymbol{\eta}_i \right| \end{aligned}$$

and the inequality (D.14) is obtained due to the fact that

$$\log\left(\frac{1}{\sigma_i^*}\right) - \log\sqrt{2\pi} \leq \log M \Rightarrow \left|\frac{1}{\sigma_i^*}\right| \leq \sqrt{2\pi} M$$

because

$$L_i\left(\boldsymbol{\eta}_i^*, \sigma_i^*\right) = \gamma_i(-\log\sigma_i^* - \log\sqrt{2\pi})$$

and

$$L_i\left(\boldsymbol{\eta}_i^*, \sigma_i^*\right) \leq \gamma_i \log M$$

since $N(s_{t_j}; \mathbf{u}_{t_j}^{\mathrm{T}} \boldsymbol{\eta}_i^*, \sigma_i^*) \leq M$ for any $j \leq i \leq n$.

Substituting inequalities (D.13) and (D.14) in (D.12), we have that

$$L_i\left(\boldsymbol{\eta}_i^*, \sigma_i^*\right) - L_i\left(\boldsymbol{\eta}_i, \sigma_i\right) \leq \pi M^2 \lambda^i \left\|\boldsymbol{\eta}_i^*\right\| \left\|\boldsymbol{\eta}_i\right\|$$

that leads to bound in (4.11) using the definition of $\boldsymbol{\eta}_i$ given by (D.3) and the following inequalities

$$\left\| \mathbf{P}_i \sum_{j=1}^{i} \lambda^{i-j} \mathbf{u}_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \boldsymbol{\eta}_i^* \right\| \leq \left\| \mathbf{P}_i \sum_{j=1}^{i} \lambda^{i-j} \mathbf{u}_{t_j} \mathbf{u}_{t_j}^{\mathrm{T}} \right\| \left\|\boldsymbol{\eta}_i^*\right\| \leq \left\|\boldsymbol{\eta}_i^*\right\|$$

where the last inequality is obtained because for any i such that the matrix (4.6) is not singular, we have that

$$\left(\sum_{j=1}^{i}\lambda^{i-j}\mathbf{u}_{t_j}\mathbf{u}_{t_j}^{\mathrm{T}}\right)^{-1}\left(\sum_{j=1}^{i}\lambda^{i-j}\mathbf{u}_{t_j}\mathbf{u}_{t_j}^{\mathrm{T}}+\lambda^{i}\mathbf{I}_K\right)\succeq\mathbf{I}_K$$

which implies

$$\left(\sum_{j=1}^{i}\lambda^{i-j}\mathbf{u}_{t_j}\mathbf{u}_{t_j}^{\mathrm{T}}+\lambda^{i}\mathbf{I}_K\right)^{-1}\sum_{j=1}^{i}\lambda^{i-j}\mathbf{u}_{t_j}\mathbf{u}_{t_j}^{\mathrm{T}}=\mathbf{P}_i\sum_{j=1}^{i}\lambda^{i-j}\mathbf{u}_{t_j}\mathbf{u}_{t_j}^{\mathrm{T}}\preceq\mathbf{I}_K.$$

Now, we proof by induction that for any $i\geq i_0$ parameters $\boldsymbol{\eta}_i$ and σ_i given by recursions (4.7)-(4.10) with $\boldsymbol{\eta}_{i_0}$, σ_{i_0}, $\mathbf{P}_{i_0}$, and γ_{i_0} given by (4.12)-(4.15) satisfy $\boldsymbol{\eta}_i=\boldsymbol{\eta}_i^*$, $\sigma_i=\sigma_i^*$, and $\mathbf{P}_i=(\sum_{j=1}^{i}\lambda^{i-j}\mathbf{u}_{t_j}\mathbf{u}_{t_j}^{\mathrm{T}})^{-1}$. Firstly, for $i=i_0$ the assertions are obtained directly from (4.12)-(4.15) since $\mathbf{H}_{i_0}$ is non-singular and $\boldsymbol{\eta}_i^*$ and σ_i^* satisfy D.1 and D.2, respectively.

If $\boldsymbol{\eta}_{i-1}=\boldsymbol{\eta}_{i-1}^*$ and $\sigma_{i-1}=\sigma_{i-1}^*$ hold, then for i we have that

$$\mathbf{P}_i=(\lambda\mathbf{P}_{i-1}^{-1}+\mathbf{u}_{t_i}\mathbf{u}_{t_i}^{\mathrm{T}})^{-1}=\left(\sum_{j=1}^{i}\lambda^{i-j}\mathbf{u}_{t_j}\mathbf{u}_{t_j}^{\mathrm{T}}\right)^{-1}\tag{D.15}$$

applying the matrix inversion Lemma to equation (4.9). From (4.7), we get

$$\begin{aligned}\boldsymbol{\eta}_i=&\mathbf{P}_{i-1}\mathbf{q}_{i-1}+\frac{\mathbf{P}_{i-1}\mathbf{u}_{t_i}}{\lambda+\mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\left(s_{t_i}-\mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{q}_{i-1}\right)\\=&\frac{\mathbf{P}_{i-1}}{\lambda+\mathbf{u}_{t_i}^{\mathrm{T}}\mathbf{P}_{i-1}\mathbf{u}_{t_i}}\mathbf{q}_i=\mathbf{P}_i\mathbf{q}_i\end{aligned}$$

by replacing the induction hypothesis and using (D.15) together with the matrix inversion Lemma. Then, the result for $\sigma_i=\sigma_i^*$ can be obtained analogously to the steps in (D.9)-(D.11). □

D.2 Proof of Theorem 13

The proof uses the following Lemma.

Lemma. Let $N(x;a,b)$, $N(y;\alpha x,\beta)$ be two Gaussian density functions, then

$$\begin{aligned}&N(x;a,b)\,N(y;\alpha x,\beta)\\&=N\left(x;\frac{a\beta^2+\alpha y b^2}{\beta^2+\alpha^2b^2},\sqrt{\frac{b^2\beta^2}{\beta^2+\alpha^2b^2}}\right)N\left(y;a\alpha,\sqrt{\beta^2+\alpha^2b^2}\right).\end{aligned}$$

Proof.

$$\begin{aligned} N(x;a,b)\,N(y;\alpha x,\beta) =& \frac{1}{2\pi b\beta}\exp\left\{\frac{-(x-a)^2}{2b^2}+\frac{-(y-\alpha x)^2}{2\beta^2}\right\} \\ =& \frac{1}{2\pi b\beta}\exp\left\{-\frac{x^2-2x\frac{a\beta^2+\alpha y b^2}{\beta^2+\alpha^2b^2}+\frac{a^2\beta^2+y^2b^2}{\beta^2+\alpha^2b^2}}{2\frac{b^2\beta^2}{\beta^2+\alpha^2b^2}}\right\}. \end{aligned}$$

Then, the result is obtained since the above expression equals

$$\frac{1}{2\pi b\beta}\exp\left\{-\frac{\left(x-\frac{a\beta^2+\alpha y b^2}{\beta^2+\alpha^2b^2}\right)^2}{2\frac{b^2\beta^2}{\beta^2+\alpha^2b^2}}-\frac{(y-\alpha a)^2}{2\left(\beta^2+\alpha^2b^2\right)}\right\}$$

by completing the squares. □

Proof of Theorem 13.

In the following, $s_{t:t+i}$ and $\mathbf{r}_{t+1:t+i}$ denote the sequences $\{s_t, s_{t+1}, ..., s_{t+i}\}$ and $\{\mathbf{r}_{t+1}, \mathbf{r}_{t+2}, ..., \mathbf{r}_{t+i}\}$ respectively, for any i.

We proceed by induction, for $i = 1$ we have that

$$\begin{aligned} p(s_{t+1}|s_t,\mathbf{r}_{t+1}) &\propto p(s_{t+1},s_t,\mathbf{r}_{t+1}) = p(\mathbf{r}_{t+1}|s_{t+1},s_t)p(s_{t+1}|s_t)p(s_t) \\ &\propto p(\mathbf{r}_{t+1}|s_{t+1})\,p(s_{t+1}|s_t) && \text{(D.16)} \\ &\propto N\left(s_{t+1};\mathbf{u}_r^{\mathrm{T}}\boldsymbol{\eta}_{r,c},\sigma_{r,c}\right)N\left(s_{t+1};\mathbf{u}_s^{\mathrm{T}}\boldsymbol{\eta}_{s,c},\sigma_{s,c}\right) && \text{(D.17)} \end{aligned}$$

where proportionalty relationships are due to the fact that s_t and $\mathbf{r}_{t+1}$ are known, (D.16) is obtained because the conditional distribution of $\mathbf{r}_{t+1}$ depends only on s_{t+1} since $\{s_t, \mathbf{r}_t\}_{t\geq 1}$ form a HMM, and (D.17) is obtained because we model conditional distributions as Gaussian given by (4.2) and (4.3).

Using the previous Lemma, (D.17) leads to (4.17) with $\hat{s}_{t+1}$ and $\hat{e}_{t+1}$ given by (4.18) and (4.19), respectively, since $\hat{s}_t = s_t$ and $\hat{e}_t = 0$.

If the statements hold for $i-1$, then for i we have that

$$\begin{aligned} &p(s_{t+i}|s_t,\mathbf{r}_{t+1:t+i}) \propto p(s_{t+i},s_t,\mathbf{r}_{t+1:t+i}) \\ &= \int p(s_t,s_{t+i-1:t+i},\mathbf{r}_{t+1:t+i})\,ds_{t+i-1} && \text{(D.18)} \\ &= \int p(s_t,s_{t+i-1:t+i},\mathbf{r}_{t+1:t+i-1})\,p(\mathbf{r}_{t+i}|s_{t+i})\,ds_{t+i-1} && \text{(D.19)} \\ &= p(\mathbf{r}_{t+i}|s_{t+i})\int p(s_t,s_{t+i-1},\mathbf{r}_{t+1:t+i-1})\,p(s_{t+i}|s_{t+i-1})\,ds_{t+i-1} && \text{(D.20)} \\ &\propto p(\mathbf{r}_{t+i}|s_{t+i})\int p(s_{t+i-1}|s_t,\mathbf{r}_{t+1:t+i-1})\,p(s_{t+i}|s_{t+i-1})\,ds_{t+i-1} \\ &\propto N\left(s_{t+i};\mathbf{u}_r^{\mathrm{T}}\boldsymbol{\eta}_{r,c},\sigma_{r,c}\right) && \text{(D.21)} \\ &\quad\cdot\int N\left(s_{t+i-1};\hat{s}_{t+i-1},\hat{e}_{t+i-1}\right)N\left(s_{t+i};\mathbf{u}_s^{\mathrm{T}}\boldsymbol{\eta}_{s,c},\sigma_{s,c}\right)ds_{t+i-1} \end{aligned}$$

where proportionalty relationships are due to the fact that s_t and $\mathbf{r}_{t+1:t+i}$ are known, (D.18) is obtained by marginalizing, (D.19) and (D.20) are obtained by using the properties of HMMs, and (D.21) is obtained by using the induction hypothesis and the models of conditional distributions as Gaussians given by (4.2) and (4.3). Then, the result is obtained by applying the previous Lemma to (D.21) twice, and substituting $\mathbf{u}_s = [1, s_{t+i-1}]^{\mathrm{T}}$. ■

Bibliography

[1] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems*, pages 1041–1048, 2009.

[2] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1):151–175, 2010.

[3] Xi Lin, Zhiyuan Yang, Qingfu Zhang, and Sam Kwong. Controllable Pareto Multi-Task Learning. In *Advances in Neural Information Processing Systems*, 2020.

[4] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[5] Lijun Zhang, Shiyin Lu, and Zhi-Hua Zhou. Adaptive online learning in dynamic environments. In *International Conference on Neural Information Processing Systems*, pages 1330–1340, 2018.

[6] Ashraf Tahmasbi, Ellango Jothimurugesan, Srikanta Tirthapura, and Phillip B Gibbons. DriftSurf: stable-state/reactive-state learning under concept drift. In *International Conference on Machine Learning*, pages 10054–10064. PMLR, 2021.

[7] Paul Ruvolo and Eric Eaton. ELLA: An efficient lifelong learning algorithm. In *International Conference on Machine Learning*, pages 507–515. PMLR, 2013.

[8] Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207, 2018.

[9] Petar Stojanov, Mingming Gong, Jaime G. Carbonell, and Kun Zhang. Data-driven approach to multiple-source domain adaptation. In *International Conference on Artificial Intelligence and Statistics*, volume 89, 2020.

[10] Hao Zheng, Ruili Wang, Wanting Ji, Ming Zong, Wai Keung Wong, Zhihui Lai, and Hexin Lv. Discriminative deep multi-task learning for facial expression recognition. *Information Sciences*, 533:60–71, 2020.

[11] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30, pages 6467–6476, 2017.

[12] Ryo Karakida and Shotaro Akaho. Learning curves for continual learning in neural networks: Self-knowledge transfer and forgetting. In *International Conference on Learning Representations*, 2022.

[13] Pengsheng Guo, Chen-Yu Lee, and Daniel Ulbricht. Learning to branch for multi-task learning. pages 3854–3863. PMLR, 2020.

[14] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[15] Julio Hurtado, Alain Raymond, and Alvaro Soto. Optimizing reusable knowledge for continual learning via metalearning. 2021.

[16] Christian Henning, Maria Cervera, Francesco D'Angelo, Johannes Von Oswald, Regina Traber, Benjamin Ehret, Seijin Kobayashi, Benjamin F Grewe, and João Sacramento. Posterior meta-replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[17] Gregory Ditzler, Manuel Roveri, Cesare Alippi, and Robi Polikar. Learning in nonstationary environments: A survey. *IEEE Computational Intelligence Magazine*, 10(4):12–25, 2015.

[18] Sarah Jane Delany, Pádraig Cunningham, Alexey Tsymbal, and Lorcan Coyle. A case-based technique for tracking concept drift in spam filtering. *Knowledge-Based Systems*, 4(18):187–195, 2005.

[19] Shiry Ginosar, Kate Rakelly, Sarah Sachs, Brian Yin, and Alexei A Efros. A century of portraits: A visual historical record of american high school yearbooks. *IEEE International Conference on Computer Vision Workshops*, 2015.

[20] Haixun Wang, Wei Fan, Philip S Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 226–235, 2003.

[21] Henry W.J. Reeve, Timothy I. Cannings, and Richard J. Samworth. Adaptive Transfer Learning. *Annals of Statistics*, 49(6):3618–3649, 2021.

[22] Yi Yao and Gianfranco Doretto. Boosting for transfer learning with multiple sources. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, number December, pages 1855–1862, 2010.

[23] Nilesh Tripuraneni, Michael I. Jordan, and Chi Jin. On the theory of transfer learning: The importance of task diversity. In *Advances in Neural Information Processing Systems*, 2020.

[24] Lixin Duan, Ivor W Tsang, Dong Xu, and Tat-Seng Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *International Conference on Machine Learning*, pages 289–296, 2009.

[25] Sicheng Zhao, Guangzhi Wang, Shanghang Zhang, Yang Gu, Yaxian Li, Zhichao Song, Pengfei Xu, Runbo Hu, Hua Chai, and Kurt Keutzer. Multi-source distilling domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12975–12983, 2020.

[26] Jafar Tahmoresnezhad and Sattar Hashemi. Visual domain adaptation via transfer feature learning. *Knowledge and information systems*, 50:585–605, 2017.

[27] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International conference on machine learning*, pages 10–18. PMLR, 2013.

[28] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International Conference on Machine Learning*, pages 1989–1998. Pmlr, 2018.

[29] Jun Wen, Risheng Liu, Nenggan Zheng, Qian Zheng, Zhefeng Gong, and Junsong Yuan. Exploiting local feature patterns for unsupervised domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5401–5408, 2019.

[30] Theodoras Evgeniou and Massimiliano Pontil. Regularized multi-task learning. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117, 2004.

[31] Yu Zhang and Qiang Yang. An overview of multi-task learning. *National Science Review*, 5(1):30–43, 2018.

[32] Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, pages 845–850, 2015.

[33] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3994–4003, 2016.

[34] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.

[35] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Philip S Yu. Learning multiple tasks with multilinear relationship networks. *Advances in neural information processing systems*, 30, 2017.

[36] Di Zhou, Jun Wang, Bin Jiang, Hua Guo, and Yajun Li. Multi-task multi-view learning based on cooperative multi-objective optimization. *IEEE Access*, 6:19465–19477, 2017.

[37] Massimiliano Pontil and Andreas Maurer. Excess risk bounds for multitask learning with trace norm regularization. In *Conference on Learning Theory*, pages 55–76. PMLR, 2013.

[38] Chuang Lin, Sicheng Zhao, Lei Meng, and Tat-Seng Chua. Multi-source domain adaptation for visual sentiment classification. In *AAAI Conference on Artificial Intelligence*, volume 34, pages 2661–2668, 2020.

[39] Hal Daumé III. Frustratingly easy domain adaptation. *ACL 2007*, page 256, 2007.

[40] Aleix M Martinez. Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class. *IEEE Transactions on Pattern analysis and machine intelligence*, 24(6):748–763, 2002.

[41] AAM Muzahid, Wanggen Wan, Ferdous Sohel, Lianyao Wu, and Li Hou. Curvenet: Curvature-based multitask learning deep networks for 3d object recognition. *IEEE/CAA Journal of Automatica Sinica*, 8(6):1177–1187, 2020.

[42] Yu Kong, Ming Shao, Kang Li, and Yun Fu. Probabilistic low-rank multitask learning. *IEEE Transactions on Neural Networks and Learning Systems*, 29(3):670–680, 2017.

[43] Jesse Read, Albert Bifet, Bernhard Pfahringer, and Geoff Holmes. Batch-incremental versus instance-incremental learning in dynamic and evolving data. *International symposium on intelligent data analysis*, 2012.

[44] Ryan Elwell and Robi Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10):1517–1531, 2011.

[45] Dariusz Brzezinski and Jerzy Stefanowski. Reacting to different types of concept drift: The accuracy updated ensemble algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1), 2013.

[46] Francesco Orabona, Joseph Keshet, and Barbara Caputo. The projectron: a bounded kernel-based perceptron. In *International Conference on Machine Learning*, pages 720–727, 2008.

[47] Jyrki Kivinen, Alexander J Smola, and Robert C Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52(8):2165–2176, 2004.

[48] Yanning Shen, Tianyi Chen, and Georgios B Giannakis. Random feature-based online multi-kernel learning in environments with unknown dynamics. *The Journal of Machine Learning Research*, 20(1):773–808, 2019.

[49] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2018.

[50] Giulia Denevi, Dimitris Stamos, Carlo Ciliberto, and Massimiliano Pontil. Online-within-online meta learning. In *Advances in Neural Information Processing Systems*, volume 32, pages 1–11, 2019.

[51] Ju Xu and Zhanxing Zhu. Reinforced continual learning. *Advances in Neural Information Processing Systems*, 31, 2018.

[52] Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. *arXiv preprint arXiv:1901.11356*, 2019.

[53] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[54] Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information Fusion*, 58:52–68, 2020.

[55] Mahmudul Hasan and Amit K Roy-Chowdhury. A continuous learning framework for activity recognition using deep hybrid feature models. *IEEE Transactions on Multimedia*, 17(11):1909–1922, 2015.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[57] Albert Bifet and Ricard Gavalda. Learning from time-changing data with adaptive windowing. In *SIAM International Conference on Data Mining*, pages 443–448. SIAM, 2007.

[58] Thi Thu Thuy Nguyen, Tien Thanh Nguyen, Alan Wee-Chung Liew, and Shi-Lin Wang. Variational inference based bayes online classifiers with concept drift adaptation. *Pattern Recognition*, 81:280–293, 2018.

[59] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision*, 2018.

[60] Jyrki Kivinen, Alex J Smola, and Robert C Williamson. Online learning with kernels. In *Proceedings of the 14th International Conference on Neural Information Processing Systems*, pages 785–792, 2001.

[61] Giovanni Cavallanti, Nicolo Cesa Bianchi, and Claudio Gentile. Tracking the best hyperplane with a simple budget perceptron. *Machine Learning*, 69(2-3):143–167, 2007.

[62] Ralf Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis*, 8(3):281–300, 2004.

[63] Nicos G Pavlidis, Dimitris K Tasoulis, Niall M Adams, and David J Hand. λ-perceptron: An adaptive classifier for data streams. *Pattern Recognition*, 44(1), 2011.

[64] Anastasia Pentina and Christoph H Lampert. Lifelong learning with non-iid tasks. In *Advances in Neural Information Processing Systems*, volume 28, pages 1540–1548, 2015.

[65] Han Zhao, Shanghang Zhang, Guanhang Wu, João P. Costeira, José M.F. Moura, and Geoffrey J. Gordon. Adversarial multiple source domain adaptation. In *Advances in Neural Information Processing Systems*, pages 8559–8570, 2018.

[66] Pierre Alquier, Massimiliano Pontil, et al. Regret bounds for lifelong learning. In *Artificial Intelligence and Statistics*, pages 261–269. PMLR, 2017.

[67] Chuan Xian Ren, Yong Hui Liu, Xi Wen Zhang, and Ke Kun Huang. Multi-source unsupervised domain adaptation via pseudo target domain. *IEEE Transactions on Image Processing*, 31, 2022.

[68] Ron Amit and Ron Meir. Meta-learning by adjusting priors based on extended pac-bayes theory. In *International Conference on Machine Learning*, pages 205–214. PMLR, 2018.

[69] Alec Farid and Anirudha Majumdar. Generalization bounds for meta-learning via PAC-Bayes and uniform stability. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[70] Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. Meta-learning without memorization. In *International Conference on Learning Representations*, 2020.

[71] Philip M Long. The complexity of learning according to two models of a drifting environment. *Machine Learning*, 37(3):337–354, 1999.

[72] Mehryar Mohri and Andres Munoz Medina. New analysis and algorithm for learning with drifting distributions. In *International Conference on Algorithmic Learning Theory*, pages 124–138, 2012.

[73] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

[74] Jing Lu, Steven CH Hoi, Jialei Wang, Peilin Zhao, and Zhi-Yong Liu. Large scale online kernel learning. *The Journal of Machine Learning Research*, 17(1):1613–1655, 2016.

[75] Tu Dinh Nguyen, Trung Le, Hung Bui, and Dinh Phung. Large-scale online kernel learning with random feature reparameterization. In *International Joint Conference on Artificial Intelligence*, pages 2543–2549, 2017.

[76] Santiago Mazuelas, Andrea Zanoni, and Aritz Perez. Minimax classification with 0-1 loss and performance guarantees. In *Advances in Neural Information Processing Systems*, pages 302–312, 2020.

[77] Santiago Mazuelas, Mauricio Romero, and Peter Grünwald. Minimax risk classifiers with 0-1 loss. *arXiv preprint*, arXiv:2201.06487, 2022.

[78] Santiago Mazuelas, Yuan Shen, and Aritz Pérez. Generalized maximum entropy for supervised classification. *IEEE Transactions on Information Theory*, 68(4):2530–2550, 2022.

[79] Yu Nesterov and Vladimir Shikhman. Quasi-monotone subgradient methods for nonsmooth convex minimization. *Journal of Optimization Theory and Applications*, 165(3):917–940, 2015.

[80] Wei Tao, Zhisong Pan, Gaowei Wu, and Qing Tao. The strength of Nesterov's extrapolation in the individual convergence of nonsmooth optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 31(7):2557–2568, 2019.

[81] Anders Isaksson, Mikael Wallman, Hanna Göransson, and Mats G Gustafsson. Cross-validation and bootstrapping are unreliable in small sample classification. *Pattern Recognition Letters*, 29(14):1960–1965, 2008.

[82] Gaël Varoquaux. Cross-validation failure: Small sample sizes lead to large error bars. *Neuroimage*, 180:68–77, 2018.

[83] Anastasia Pentina and Christoph Lampert. A PAC-Bayesian bound for lifelong learning. In *International Conference on Machine Learning*, pages 991–999. PMLR, 2014.

[84] Anastasia Pentina and Ruth Urner. Lifelong learning with weighted majority votes. In *Advances in Neural Information Processing Systems*, volume 29, pages 3612–3620, 2016.

[85] Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.

[86] Liangwei Zhang, Jing Lin, and Ramin Karim. Sliding window-based fault detection from high-dimensional data streams. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(2):289–303, 2016.

[87] Honorius Gâlmeanu and Răzvan Andonie. Weighted incremental-decremental support vector machines for concept drift with shifting window. *Neural Networks*, 2022.

[88] Morvarid Karimpour, Shiva Noori Saray, Jafar Tahmoresnezhad, and Mohammad Pourmahmood Aghababa. Multi-source domain adaptation for image classification. *Machine Vision and Applications*, 31(6):1–19, 2020.

[89] Kun Zhang, Mingming Gong, and Bernhard Scholkopf. Multi-source domain adaptation: A causal view. In *National Conference on Artificial Intelligence*, volume 4, pages 3150–3157, 2015.

[90] Keqiuyin Li, Jie Lu, Hua Zuo, and Guangquan Zhang. Multi-Source Contribution Learning for Domain Adaptation. *IEEE Transactions on Neural Networks and Learning Systems*, 33(10):5293–5307, 2022.

[91] J Zico Kolter and Marcus A Maloof. Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research*, 8(12):2755–2790, 2007.

[92] Sérgio Jesus, José Pombal, Duarte Alves, André Cruz, Pedro Saleiro, Rita P. Ribeiro, João Gama, and Pedro Bizarro. Turning the Tables: Biased, Imbalanced, Dynamic Tabular Datasets for ML Evaluation. In *Advances in Neural Information Processing Systems*, 2022.

[93] Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Philipp Kranen, Hardy Kremer, Timm Jansen, and Thomas Seidl. MOA: Massive online analysis, a framework for stream classification and clustering. In *Proceedings of the First Workshop on Applications of Pattern Analysis*, pages 44–50. PMLR, 2010.

[94] Ofer Dekel, Shai Shalev-Shwartz, and Yoram Singer. The forgetron: A kernel-based perceptron on a budget. *SIAM Journal on Computing*, 37(5):1342–1372, 2008.

[95] Tegjyot Singh Sethi and Mehmed Kantardzic. On the reliable detection of concept drift from streaming unlabeled data. *Expert Systems with Applications*, 82:77–99, 2017.

[96] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

[97] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469, 2022.

[98] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. *IEEE International Conference on Computer Vision*, 2019.

[99] Zhifei Zhang, Yang Song, and Hairong Qi. Age progression/regression by conditional adversarial autoencoder. *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[100] Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. The CLEAR benchmark: Continual learning on real-world imagery. In *Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[101] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), 2013.

[102] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[103] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. In *International Conference on Learning Representations*, 2018.

[104] Peng Zhao, Le-Wen Cai, and Zhi-Hua Zhou. Handling concept drift via model reuse. *Machine Learning*, 109(3):533–568, 2020.

[105] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey:

Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[106] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys (CSUR)*, 46(4):1–37, 2014.

[107] Geoffrey I Webb, Loong Kuan Lee, Bart Goethals, and François Petitjean. Analyzing concept drift and shift from sample data. *Data Mining and Knowledge Discovery*, 32(5):1179–1199, 2018.

[108] Ashok Cutkosky. Parameter-free, dynamic, and strongly-adaptive online learning. In *International Conference on Machine Learning*, pages 2250–2259, 2020.

[109] Yaakov Bar Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[110] Brian J Odelson, Murali R Rajamani, and James B Rawlings. A new autocovariance least-squares method for estimating noise covariances. *Automatica*, 42(2):303–308, 2006.

[111] Shahrokh Akhlaghi, Ning Zhou, and Zhenyu Huang. Adaptive adjustment of noise covariance in Kalman filter for dynamic state estimation. In *IEEE Power & Energy Society General Meeting*, pages 1–5, 2017.

[112] Jeffrey Humpherys, Preston Redd, and Jeremy West. A fresh look at the Kalman filter. *SIAM review*, 54(4):801–823, 2012.

[113] Atsutoshi Kumagai and Tomoharu Iwata. Learning future classifiers without additional data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[114] Atsutoshi Kumagai and Tomoharu Iwata. Learning non-linear dynamics of decision boundaries for maintaining classification performance. In *AAAI Conference on Artificial Intelligence*, 2017.

[115] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[116] Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. Achieving forgetting prevention and knowledge transfer in continual learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[117] Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[118] Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

[119] Verónica Álvarez, Santiago Mazuelas, and Jose A Lozano. Minimax classification under concept drift with multidimensional adaptation and performance guarantees. *International Conference on Machine Learning*, 2022.

[120] Brian DO Anderson and John B Moore. *Optimal Filtering*. Courier Corporation, 2012.

[121] Xisen Jin, Arka Sadhu, Junyi Du, and Xiang Ren. Gradient-based editing of memory examples for online task-free continual learning. *Advances in Neural Information Processing Systems*, 2021.

[122] H Lee Willis. *Power distribution planning reference book*. CRC press, 2004.

[123] Rafal Weron. *Modeling and forecasting electricity loads and prices: A statistical approach*, volume 403. John Wiley & Sons, 2007.

[124] Damitha K Ranaweera, George G Karady, and Richard G Farmer. Economic impact analysis of load forecasting. *IEEE Transactions on Power Systems*, 12(3):1388–1392, 1997.

[125] Muhammad Usman Fahad and Naeem Arbab. Factor affecting short term load forecasting. *Journal of Clean Energy Technologies*, 2(4):305–309, 2014.

[126] Samsher Kadir Sheikh and MG Unde. Short term load forecasting using ANN technique. *International Journal of Engineering Sciences & Emerging Technologies*, 1(2):97–107, 2012.

[127] J. W. Taylor and R. Buizza. Neural network load forecasting with weather ensemble predictions. *IEEE Transactions on Power Systems*, 17(3):626–632, 2002.

[128] Marcelo Espinoza, Johan AK Suykens, Ronnie Belmans, and Bart De Moor. Electric load forecasting. *IEEE Control Systems Magazine*, 27(5):43–57, 2007.

[129] Gesche Huebner, David Shipworth, Ian Hamilton, Zaid Chalabi, and Tadj Oreszczyn. Understanding electricity consumption: A comparative contribution of building factors, socio-demographics, appliances, behaviours and attitudes. *Applied Energy*, 177:692–702, 2016.

[130] Tobias Kronenberg. The impact of demographic change on energy use and greenhouse gas emissions in Germany. *Ecological Economics*, 68(10):2637–2645, 2009.

[131] Nima Amjady, Farshid Keynia, and Hamidreza Zareipour. Short-term load forecast of microgrids by a new bilevel prediction strategy. *IEEE Transactions on Smart Grid*, 1(3):286–294, 2010.

[132] Fahad Javed, Naveed Arshad, Fredrik Wallin, Iana Vassileva, and Erik Dahlquist. Forecasting for demand response in smart grids: An analysis on use of anthropologic and structural data and short term multiple loads forecasting. *Applied Energy*, 96:150–160, 2012.

[133] Weicong Kong, Zhao Yang Dong, David J Hill, Fengji Luo, and Yan Xu. Short-term residential load forecasting based on resident behaviour learning. *IEEE Transactions on Power Systems*, 33(1):1087–1088, 2017.

[134] Erol Gelenbe and Yves Caseau. The impact of information technology on energy consumption and carbon emissions. *Ubiquity*, 2015(June):1–15, 2015.

[135] Renato Baran and Luiz Fernando Loureiro Legey. The introduction of electric vehicles in Brazil: Impacts on oil and electricity consumption. *Technological Forecasting and Social Change*, 80(5):907–917, 2013.

[136] Takayuki Shiina and John R Birge. Stochastic unit commitment problem. *International Transactions in Operational Research*, 11(1):19–32, 2004.

[137] Tao Hong and Shu Fan. Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting*, 32(3):914–938, 2016.

[138] Shai Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

[139] S Sp Pappas, L Ekonomou, D Ch Karamousantas, GE Chatzarakis, SK Katsikas, and P Liatsis. Electricity demand loads modeling using auto regressive moving average (ARMA) models. *Energy*, 33(9):1353–1360, 2008.

[140] Shyh-Jier Huang and Kuang-Rong Shih. Short-term load forecasting via ARMA model identification including non-Gaussian process considerations. *IEEE Transactions on Power Systems*, 18(2):673–679, 2003.

[141] Nima Amjady. Short-term hourly load forecasting using time-series modeling with peak load estimation capability. *IEEE Transactions on Power Systems*, 16(3):498–505, 2001.

[142] Norizan Mohamed, Maizah Hura Ahmad, and Zuhaimy Ismail. Short term load forecasting using double seasonal ARIMA model. In *Proceedings of the Regional Conference on Statistical Sciences*, volume 10, pages 57–73, 2010.

[143] Yacine Chakhchoukh, Patrick Panciatici, and Lamine Mili. Electric load forecasting based on statistical robust methods. *IEEE Transactions on Power Systems*, 26(3):982–991, 2010.

[144] Kunjin Chen, Kunlong Chen, Qin Wang, Ziyu He, Jun Hu, and Jinliang He. Short-term load forecasting with deep residual networks. *IEEE Transactions on Smart Grid*, 10(4):3943–3952, 2018.

[145] Henrique Steinherz Hippert, Carlos Eduardo Pedreira, and Reinaldo Castro Souza. Neural networks for short-term load forecasting: A review and evaluation. *IEEE Transactions on Power Systems*, 16(1):44–55, 2001.

[146] Bo-Juen Chen and Ming-Wei Chang. Load forecasting using support vector machines: A study on EUNITE competition 2001. *IEEE Transactions on Power Systems*, 19(4):1821–1830, 2004.

[147] Shu Fan and Luonan Chen. Short-term load forecasting based on an adaptive hybrid method. *IEEE Transactions on Power Systems*, 21(1):392–401, 2006.

[148] Yang Wang, Qing Xia, and Chongqing Kang. Secondary forecasting based on deviation analysis for short-term load forecasting. *IEEE Transactions on Power Systems*, 26(2):500–507, 2010.

[149] Tomonobu Senjyu, Paras Mandal, Katsumi Uezato, and Toshihisa Funabashi. Next day load curve forecasting using hybrid correction method. *IEEE Transactions on Power Systems*, 20(1):102–109, 2005.

[150] João Lourenço and Paulo Santos. Short term load forecasting using Gaussian process models. *Proceedings of Instituto de Engenharia de Sistemas e Computadores de Coimbra*, 2010.

[151] Bidong Liu, Jakub Nowotarski, Tao Hong, and Rafał Weron. Probabilistic load forecasting via quantile regression averaging on sister forecasts. *IEEE Transactions on Smart Grid*, 8(2):730–737, 2015.

[152] Yi Wang, Qixin Chen, Ning Zhang, and Yishen Wang. Conditional residual modeling for probabilistic load forecasting. *IEEE Transactions on Power Systems*, 33(6):7327–7330, 2018.

[153] Larry D Paarmann and Mohamed D Najar. Adaptive online load forecasting via time series modeling. *Electric Power Systems Research*, 32(3):219–225, 1995.

[154] Abderrezak Laouafi, Mourad Mordjaoui, Salim Haddad, Taqiy Eddine Boukelia, and Abderahmane Ganouche. Online electricity demand forecasting based on an effective forecast combination methodology. *Electric Power Systems Research*, 148:35–47, 2017.

[155] Amadou Ba, Mathieu Sinn, Yannig Goude, and Pascal Pompey. Adaptive learning of smoothing functions: Application to electricity load forecasting. In *Advances in Neural Information Processing Systems*, pages 2510–2518, 2012.

[156] Tao Hong, Pierre Pinson, Shu Fan, Hamidreza Zareipour, Alberto Troccoli, and Rob J Hyndman. Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond. *International Journal of Forecasting*, 32(3):896–913, 2016.

[157] Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):243–268, 2007.

[158] Santiago Mazuelas, Yuan Shen, and Moe Z Win. Belief condensation filtering. *IEEE Transactions on Signal Processing*, 61(18):4403–4415, 2013.

[159] Olivier Cappé, Eric Moulines, and Tobias Rydén. *Inference in hidden Markov models*. Springer Science & Business Media, 2006.

[160] Weifeng Liu, Jose C Principe, and Simon Haykin. *Kernel adaptive filtering: a comprehensive introduction*, volume 57. John Wiley & Sons, 2011.

[161] Marcelo Espinoza, Johan AK Suykens, and Bart De Moor. Load forecasting using fixed-size least squares support vector machines. In *Computational Intelligence and Bioinspired Systems*, volume 3512, pages 1018–1026. Springer, 2005.

[162] Tao Hong, Pierre Pinson, and Shu Fan. Global energy forecasting competition 2012. *International Journal of Forecasting*, 30(2):357–363, 2014.

[163] Athanasios P Liavas and Phillip A Regalia. On the numerical stability and accuracy of the conventional recursive least squares algorithm. *IEEE Transactions on Signal Processing*, 47(1):88–96, 1999.

[164] Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, pages 3215–3225, 2017.

[165] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.