UNIVERSIDAD AUTÓNOMA DE MADRID

Advanced Kernel Methods for Multi-Task Learning

by

Carlos Ruiz Pastor

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the

Escuela Politécnica Superior

Computer Science Department

under the supervision of José R. Dorronsoro Ibero

July 2022

What is the essence of life? To serve others and to do good.

Aristotle.

Abstract

.

Resumen

.

Acknowledgements

.

Contents

Abbreviations

ADF	**A**ssumed **D**ensity **F**iltering
AF	**A**cquisition **F**unction
BO	**B**ayesian **O**ptimization
DGP	**D**eep **G**aussian **P**rocess
EI	**E**xpected **I**mprovement
EP	**E**xpectation **P**ropagation
GP	**G**aussian **P**rocess
KL	**K**ullback **L**iebler
MCMC	**M**arkov **C**hain **M**onte **C**arlo
PPESMOC	**P**arallel **P**redictive **E**ntropy **S**earch for **M**ultiobjective **O**ptimization with **C**onstraints
PES	**P**redictive **E**ntropy **S**earch
PESMOC	**P**redictive **E**ntropy **S**earch for **M**ultiobjective **O**ptimization with **C**onstraints
RS	**R**andom **S**earch
UCB	**U**pper **C**onfidence **B**ound

To my family

Chapter 1

Introduction

We begin this manuscript...

1.1 Introduction

1.2 Publications

This section presents, in chronological order, the work published during the doctoral period in which this thesis was written. We also include other research work related to this thesis, but not directly included on it. Finally, this document includes content that has not been published yet and is under revision.

Related Work

Work In Progress

1.3 Summary by Chapters

In this section...

Chapter 2 provides an introduction to GPs and the expectation propagation algorithm. Both are necessary concepts for the BO methods that we will describe in the following chapters. This chapter reviews the fundamentals of GPs and why they are so interesting for BO. More concretely, we review the most popular kernels, the analysis of the posterior and predictive distribution and how to tune the hyper-parameters of GPs: whether by maximizing the marginal likelihood or by generating samples from the hyper-parameter posterior distribution. Other alternative probabilistic surrogate models are also described briefly. Some of the proposed approaches of this thesis are extensions of an acquisition function called predictive entropy search, that is based on the expectation propagation approximate inference technique. That is why we provide in this chapter an explanation of the expectation propagation algorithm.

Chapter 5 introduces the basics of BO and information theory. BO works with probabilistic models such as GPs and with acquisition functions such as predictive entropy search, that uses information theory. Having studied GPs in Chapter 2, BO can be now understood and it is described in detail. This chapter will also

describe the most popular acquisition functions, how information theory can be applied in BO and why BO is useful for the hyper-parameter tuning of machine learning algorithms.

Chapter 4 describes an information-theoretical mechanism that generalizes BO to simultaneously optimize multiple objectives under the presence of several constraints. This algorithm is called predictive entropy search for multi-objective BO with constraints (PESMOC) and it is an extension of the predictive entropy search acquisition function that is described in Chapter 5. The chapter compares the empirical performance of PESMOC with respect to a state-of-the-art approach to constrained multi-objective optimization based on the expected improvement acquisition function. It is also compared with a random search through a set of synthetic, benchmark and real experiments.

Chapter ?? addresses the problem that faces BO when not only one but multiple input points can be evaluated in parallel that has been described in Section **??**. This chapter introduces an extension of PESMOC called parallel PESMOC (PPESMOC) that adapts to the parallel scenario. PPESMOC builds an acquisition function that assigns a value for each batch of points of the input space. The maximum of this acquisition function corresponds to the set of points that maximizes the expected reduction in the entropy of the Pareto set in each evaluation. Naive adaptations of PESMOC and the method based on expected improvement for the parallel scenario are used as a baseline to compare their performance with PPESMOC. Synthetic, benchmark and real experiments show how PPESMOC obtains an advantage in most of the considered scenarios. All the mentioned approaches are described in detail in this chapter.

Chapter ?? addresses a transformation that enables standard GPs to deliver better results in problems that contain integer-valued and categorical variables. We can apply BO to problems where we need to optimize functions that contain integer-valued and categorical variables with more guarantees of obtaining a solution with low regret. A critical advantage of this transformation, with respect to other approaches, is that it is compatible with any acquisition function. This transformation makes the uncertainty given by the GPs in certain areas of the space flat. As a consequence, the acquisition function can also be flat in these zones. This phenomenom raises an issue with the optimization of the acquisition function, that must consider the flatness of these areas. We use a one exchange neighbourhood approach to optimize the resultant acquisition function. We test our approach in synthetic and real problems, where we add empirical evidence of the performance of our proposed transformation.

Chapter ?? shows a real problem where BO has been applied with success. In this problem, BO has been used to obtain the optimal parameters of a hybrid Grouping Genetic Algorithm for attribute selection. This genetic algorithm is combined with an Extreme Learning Machine (GGA-ELM) approach for prediction of ocean wave features. Concretely, the significant wave height and the wave energy flux at a goal marine structure facility on the Western Coast of the USA is predicted. This chapter illustrates the experiments where it is shown that BO improves the performance of the GGA-ELM approach. Most importantly, it also outperforms a random search of the hyper-parameter space and the human expert criterion.

Chapter ?? provides a summary of the work done in this thesis. We include the conclusions retrieved by the multiple research lines covered in the chapters. We also illustrate lines for future research.

1.4 Definitions and Notation

Chapter 2

Foundations and Concepts

This chapter presents...

2.1 Introduction

2.2 Kernels

2.2.1 Motivation and Definition

2.2.2 Reproducing Kernel Hilbert Spaces

2.2.3 Examples and Properties

2.3 Risk Functions and Regularization

2.3.1 Empirical and Expected Risk

2.3.2 Regularized Risk Functional

2.3.3 Representer Theorem

2.4 Optimization

2.4.1 Convex Optimization

2.4.2 Unconstrained Problems

2.4.3 Constrained Problems

2.5 Statistical Learning

2.5.1 Uniform Convergence and Consistency

2.5.2 VC dimension and Structural Learning

2.6 Support Vector Machines

2.6.1 Linearly Separable Case

2.6.2 Non-Linearly Separable Case

2.6.3 Kernel Extension

2.6.4 SVM properties

2.6.5 Connection with Structural Learning

2.6.6 SVM Variants

2.7 Conclusions

In this chapter, we covered...

Chapter 3

Multi-Task Learning

This chapter presents...

3.1 Introduction

3.2 Why does Multi-Task Learning work?

TODO: Repasar sección entera después escribir capítulo 2

3.2.1 Multi-Task Learning and Learning to Learn

Typically in Machine Learning the goal is to find the best hypothesis $h(x, \alpha_0)$ from a space of hypotheses $\mathcal{H} = \{h(x, \alpha), \alpha \in A\}$, where A is any set of parameters. This best candidate can be selected according to different inductive principles, which define a method of approximating a global function $f(x)$ from a training set: $z := \{(x_i, y_i),\ i = 1, \ldots, n\}$ where (x_i, y_i) are sampled from a distribution F. In the classical statistics we find the Maximum Likelihood approach, where the goal is to estimate the density $f(x) = P(y \mid x)$ and the hypotheses space is parametric, i.e. $\mathcal{H} = \{h(x, \alpha), \alpha \in A \subset \mathbb{R}^m\}$. The learner select the parameter α that maximizes the probability of the data given the hypothesis. Another more direct inductive principle is Empirical Risk Minimization (ERM), which is the most common one. In ERM the densities are ignored and an empirical error $\hat{R}_z(h(\cdot, \alpha))$ is minimized with the hope of minimizing the true expected error $R_F(h(\cdot, \alpha))$, which would result in a good generalization. Several models use the ERM principle to generalize from data such as Neural Networks or Support Vector Machines. These methods are designed to find a good hypothesis $h(x, \alpha)$ from a given space $\mathcal{H}$. The definition of such space $\mathcal{H}$ define the bias for these problems and its selection is crucial. If $\mathcal{H}$ does not contain any good hypothesis, the learner will not be able to learn. Also, if the hypotheses space is too large, the learning process is more difficult. The best hypotheses space we can provide is the one containing only the optimal hypothesis, but this is equivalent the original problem. When we only want to estimate a single function $f(x)$, a single-task scenario, there is no difference between learning the optimal hypotheses space (bias learning), that is selecting $\mathcal{H} = \{h^*\}$ from a family of hypotheses spaces for the problem $\arg\min_{h \in \mathcal{H}} R_F(h(\cdot, \alpha))$, and ordinary learning of the optimal hypothesis function. That is, we can consider the family of hypotheses $\mathbb{H} = \{\{h(\cdot, \alpha)\}, h(\cdot, \alpha) \in \mathcal{H}\}$ and selecting the best single-element hypotheses space is equivalent to ordinary learning. Instead, we focus on the situation where we want to solve multiple related tasks, that is, estimating

multiple functions $f_1(x), \ldots, f_T(x)$. In that case, we need a good space $\mathcal{H}$ that contains good solutions for the different tasks. In Baxter (2000) an effort is made to define the concepts needed to construct the theory about inductive bias learning or Learning to Learn, which can be seen as a generalization of strict Multi-Task Learning. This is done by defining an environment of tasks and extending the work of Vapnik (2000), which defines the capacity of space of hypothesis, Baxter defines the capacity of a family of spaces of hypothesis.

Before presenting the concepts defined for Bias Learning, and to establish an analogy to those of ordinary learning, we briefly review some statistical learning concepts.

Ordinary Learning

In the ordinary statistical learning, some theoretical concepts are used:

- an *input space* $\mathcal{X}$ and an *output space* $\mathcal{Y}$,
- a *probability distribution* F, which is unknown, defined over $\mathcal{X} \times \mathcal{Y}$,
- a *loss function* $\ell(\cdot, \cdot) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, and
- a *hypotheses space* $\mathcal{H} = \{h(x, \alpha), \alpha \in A \subset \mathbb{R}^m\}$ with hypothesis $h(\cdot, \alpha) : \mathcal{X} \to \mathcal{Y}$.

The goal for the learner is to select a hypothesis $h(x, \alpha) \in \mathcal{H}$, or equivalently $\alpha \in A$, that minimizes the expected risk

$$R_F(h(\cdot, \alpha)) = \int_{\mathcal{X} \times \mathcal{Y}} \ell(h(x, \alpha), y)\, dF(x, y).$$

The distribution F is unknown, but we have a training set $z = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ of samples drawn from F. The approach is then is to apply the ERM inductive principle, that is to minimize the empirical risk

$$\hat{R}_z(h(\cdot, \alpha)) = \frac{1}{n} \sum_{i=1}^{n} l(h(x_i), y_i).$$

Thus, a learner $\mathcal{A}$ maps the set of training samples to a set of hypotheses:

$$\mathcal{A} : \bigcup (\mathcal{X} \times \mathcal{Y})^n \to \mathcal{H}.$$

Although $\hat{R}_z(h(\cdot, \alpha))$ is an unbiased estimator of $R_F(h(\cdot, \alpha))$, it has been shown Vapnik (2000) that this approach, despite being the most evident one, is not the best principle that can be followed. This has relation with two facts: the first one is that the unbiased property is an asymptotical one, the second one has to do with overfitting. Vapnik answers to the question of what can be said about R_F when $h(\cdot, \alpha^*)$ minimizes $\hat{R}_z(h(\cdot, \alpha))$, and moreover, his results are valid also for small number of training samples n. More specifically, Vapnik sets the sufficient and necessary conditions for the consistency of an inductive learning process, i.e. for $\hat{R}_z(h(\cdot, \alpha)) \xrightarrow{P} R_F(h(\cdot, \alpha))$ uniformly. Vapnik also defines the capacity of a hypotheses space and use it to derive bounds on the rate of this convergence for any $\alpha \in A$ and, more importantly, bounds on the difference $\inf_{\alpha \in A} \hat{R}_z(h(\cdot, \alpha)) - \inf_{\alpha \in A} R_F(h(\cdot, \alpha))$. Under some general conditions, he proves that

$$\inf_{\alpha \in A} \hat{R}_z(h(\cdot, \alpha)) - \inf_{\alpha \in A} R_F(h(\cdot, \alpha)) \leq B(n/\mathrm{VCdim}(\mathcal{H})) \tag{3.1}$$

where B is some non-decreasing function and $\text{VCdim}(\mathcal{H})$ is the capacity of the space $\mathcal{H}$, also named the VC-dimension $\mathcal{H}$. This means that the generalization ability of a learning process can be controlled in terms of two factors:

- The number of training samples n. A greater number of training samples assures a better generalization of the learning process.This looks intuitive and could be already inferred from the asymptotical properties.
- The VC-dimension $\text{VCdim}(\mathcal{H})$ of the hypotheses space $\mathcal{H}$, which is desirable to be small. This term is not intuitive and is the most important term in Vapnik theory.

The VC-dimension measures the capacity of a set of hypotheses $\mathcal{H}$. If the capacity of the set $\mathcal{H}$ is too large, we may find a hypothesis $h(x, \alpha^*)$ that minimizes $\hat{R}_z$ but does not generalize well and therefore, does not minimize R_F. This is the overfitting problem. On the other side, if we use a simple $\mathcal{H}$, with low capacity, we could be in a situation where there is not a good hypothesis $h(x, \alpha) \in \mathcal{H}$, so the empirical risk $\inf_{\alpha \in A} R_F$ is too large. This is the underfitting problem.

Bias Learning: Concept and Components

In Baxter (2000) the goal is not to learn the optimal hypothesis $h(\cdot, \alpha^*)$ from a fixed space $\mathcal{H}$ but to learn a good space $\mathcal{H}$ from which we can obtain an optimal hypothesis in different situations. Two main concepts are defined: the *family of hypotheses spaces* and an *environment* of related tasks. For simplicity we write $h(x)$ instead of $h(x, \alpha)$ and h instead of $h(\cdot, \alpha)$. Using these concepts, the bias learning problem has the following components:

- an *input space* $\mathcal{X}$ and an *output space* $\mathcal{Y}$,
- an *environment* $(\mathcal{P}, Q)$ where $\mathcal{P}$ is a set of distributions P defined over $\mathcal{X} \times \mathcal{Y}$, and we can sample from $\mathcal{P}$ according to a distribution Q,
- a *loss function* $\ell(\cdot, \cdot) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, and
- a *family of hypotheses spaces* $\mathbb{H} = \{\mathcal{H}_\beta, \beta \in B\}$, where each element $\mathcal{H}_\beta$ is a set of hypotheses.

Analogous to ordinary learning, the goal is to minimize the expected risk, defined as

$$R_Q(\mathcal{H}_\beta) = \int_{\mathcal{P}} \inf_{h \in \mathcal{H}_\beta} R_P(h) dQ(P) = \int_{\mathcal{P}} \inf_{h \in \mathcal{H}_\beta} \int_{\mathcal{X} \times Y} l(h(x), y) dP(x, y) dQ(P). \tag{3.2}$$

Observe that this risk is not a function of a specific hypothesis but it depends on the selection of a whole hypotheses space $\mathcal{H}_\beta$. Again, we do not know $\mathcal{P}$ nor Q, but we have a training set samples from the environment $(\mathcal{P}, Q)$ obtained in the following way:

1. Sample T times from Q obtaining $P_1, \ldots, P_T \in \mathcal{P}$
2. For $r = 1, \ldots, T$ sample m pairs $z_r = \{(x_1^r, y_1^r), \ldots, (x_m^r, y_m^r)\}$ according to P_r where $(x_i^r, y_i^r) \in X \times Y$.

We obtain a sample $z = \{(x_i^r, y_i^r), r = 1,\ i = 1, \ldots,\ m = 1, \ldots, T\}$, with m examples from T different learning tasks, and

$$\boldsymbol{z} := \begin{matrix} (x_1^1, y_1^1) & \ldots & (x_m^1, y_m^1) \\ \vdots & \ddots & \vdots \\ (x_1^T, y_1^T) & \ldots & (x_m^T, y_m^T) \end{matrix}$$

is named as a (T, m)-sample. Using $\boldsymbol{z}$ we can define the empirical loss as

$$\hat{R}_{\boldsymbol{z}}(\mathcal{H}_\beta) = \sum_{r=1}^{T} \inf_{h \in \mathcal{H}_\beta} \hat{R}_{z_r}(h) = \sum_{r=1}^{T} \inf_{h \in \mathcal{H}_\beta} \sum_{i=1}^{m} l(h(x_i^r), y_i^r), \tag{3.3}$$

which is an average of the empirical losses of each task. Note, however, that in the case of the bias learner, this estimate is biased, since $R_{P_r}(h)$ does not coincide with $\hat{R}_{z_r}(h)$. Putting all together, a bias learner $\mathcal{A}$ maps the set of all (T, m)-samples to a family of hypotheses spaces:

$$\mathcal{A} : \bigcup (\mathcal{X} \times \mathcal{Y})^{(T,m)} \to \mathbb{H}.$$

To follow an analogous path to that of ordinary learning, the milestones in bias learning theory should include:

- Checking the consistency of the Bias Learning methods, i.e. proving that $\hat{R}_{\boldsymbol{z}}(\mathcal{H}_\beta)$ converges uniformly in probability to $R_Q(\mathcal{H}_\beta)$.
- Defining a notion of capacity of hypotheses space families $\mathbb{H}$.
- Finding a bound of $\hat{R}_{\boldsymbol{z}}(\mathcal{H}_\beta) - R_Q(\mathcal{H}_\beta)$ for any β using the capacity of the hypotheses space family. If possible, finding also a bound for $\inf_{\beta \in B} \hat{R}_{\boldsymbol{z}}(\mathcal{H}_\beta) - \inf_{\beta \in B} R_Q(\mathcal{H}_\beta)$.

To achieve these goals some previous definitions are needed. From this point, since any $\mathcal{H}$ is defined by a $\beta \in B$, we omit β and write just $\mathcal{H}$ for simplicity.

Bias Learning: Capacities and Uniform Convergence

In first place, a *sample-driven* pseudo-metric of $(T, 1)$-empirical risks is defined. Consider a sequence of T probabilities $\boldsymbol{P} = (P_1, \ldots, P_T)$ sampled from $\mathcal{P}$ according the distribution Q.

Definition 3.1 (*sample-driven* pseudometric)**.** Given a $(T, 1)$-sample

$$\left\{(x_1^1, y_1^1), (x_2^2, y_2^2), \ldots, (x_T^T, y_T^T)\right\},$$

consider the set of sequences of T hypothesis

$$\mathcal{H}^T := \{\boldsymbol{h} = (h_1, \ldots, h_T), h_1, \ldots, h_T \in \mathcal{H}\}.$$

We can define then the set of $(T, 1)$-empirical risks, with one sample per task, as

$$\mathcal{H}_\ell^T := \left\{ \boldsymbol{h}_\ell(x_1, y_1, \ldots, x_T, y_T) = \sum_{r=1}^{T} \ell(h(x_i), y_i), h_1, \ldots, h_T \in \mathcal{H} \right\}$$

The family of the set of $(T, 1)$-risks of hypothesis is then $\mathbb{H}^T = \bigcup_{\mathcal{H}\in\mathbb{H}} \mathcal{H}^T$. Now we can define

$$d_{\boldsymbol{P}}(\boldsymbol{h}_\ell, \boldsymbol{h}'_\ell) = \int_{(\mathcal{X}\times\mathcal{Y})^T} \left|\boldsymbol{h}_\ell(x_1, y_1, \ldots, x_T, y_T) - \boldsymbol{h}'_\ell(x_1, y_1, \ldots, x_T, y_T)\right| \\ dP_1(x_1, y_1) \ldots dP_T(x_T, y_T)$$

for $\boldsymbol{h}_\ell, \boldsymbol{h}'_\ell \in \mathcal{H}_\ell, \mathcal{H}'_\ell$ as a pseudo-metric in $\mathbb{H}^T$.

Then, a *distribution-driven* pseudo-metric is defined.

Definition 3.2 (*distribution-driven* pseudo-metric)**.** Given a distribution P on $\mathcal{X} \times \mathcal{Y}$. Consider the set of infimum expected risk for each $\mathcal{H}$:

$$\mathcal{H}^* := \inf_{h\in\mathcal{H}} R_P(h).$$

The family of such sets is defined as $\mathbb{H}^* = \{\mathcal{H}^*, \mathcal{H} \in \mathbb{H}\}$. The pseudo-metric in this space is given by d_Q:

$$d_Q(\mathcal{H}_1^*, \mathcal{H}_2^*) = \int_{\mathcal{P}} |\mathcal{H}_1^* - \mathcal{H}_2^*| \, dQ$$

for $\mathcal{H}_1^*, \mathcal{H}_2^* \in \mathbb{H}^*$. With these two pseudo-metrics, two capacities for families of hypotheses spaces are defined. For that the definition of ϵ-cover is needed.

Definition 3.3 (ϵ-cover)**.** Given a pseudo-metric d_S in a space $\mathcal{S}$, a set of l elements $s_1, \ldots, s_l \in \mathcal{S}$ is an ϵ-cover of $\mathcal{S}$ if $\forall s \in S \; d_S(s, s_i) \leq \epsilon$ for some $i = 1, \ldots, l$. Let $\mathcal{N}(\epsilon, \mathcal{S}, d_S)$ denote the size of the smallest ϵ-cover.

Then, we can define the following capacities of a family space $\mathbb{H}$:

- The *sample-driven capacity* $C\left(\epsilon, \mathbb{H}^T\right) := \sup_{\boldsymbol{P}} \mathcal{N}(\epsilon, \mathbb{H}^T, d_{\boldsymbol{P}})$.
- The *distribution-driven capacity* $C\left(\epsilon, \mathbb{H}^*\right) := \sup_Q \mathcal{N}(\epsilon, \mathbb{H}^*, d_Q)$.

Using these capacities, the convergence (uniformly over all $\mathcal{H} \in \mathbb{H}$) of bias learners can be proved (Baxter, 2000, Theorem 2). Moreover, the bias expected risk is bounded

$$\hat{R}_{\boldsymbol{z}}(\mathcal{H}) \leq R_Q(\mathcal{H}) + \epsilon$$

with probability $1 - \eta$, given sufficiently large T and m,

$$T \geq \max\left(\frac{256}{T\epsilon^2} \log \frac{8C\left(\frac{\epsilon}{32}, \mathbb{H}^*\right)}{\eta}, \frac{64}{\epsilon^2}\right), \; m \geq \max\left(\frac{256}{T\epsilon^2} \log \frac{8C\left(\frac{\epsilon}{32}, \mathbb{H}^T\right)}{\eta}, \frac{64}{\epsilon^2}\right).$$

It should be noted that the bound for m is inversely proportional to T, that is, the more tasks we have, the less samples we need for each task.

Multi-Task Learning

The previous result is a result for pure Bias Learning, where we have an $(\mathcal{P}, Q)$-environment of tasks. In Multi-Task Learning, we have a fixed number of tasks T and a fixed sequence of distributions $\boldsymbol{P} = (P_1, \ldots, P_T)$, where P_i is a distribution over

$(\mathcal{X} \times \mathcal{Y})^m$. The goal is not learning a hypotheses space $\mathcal{H}$ but a sequence of hypothesis $\boldsymbol{h} = (h_1, \ldots, h_T), h_1, \ldots, h_T \in \mathcal{H}$. Thus, the Multi-Task expected risk is

$$R_{\boldsymbol{P}}(\boldsymbol{h}) = \sum_{r=1}^{T} R_{P_r}(h_r) = \sum_{r=1}^{T} \int_{\mathcal{X} \times Y} l(h_r(x), y) dP_r(x, y), \tag{3.4}$$

and the empirical risk is defined as

$$\hat{R}_{\boldsymbol{z}}(\boldsymbol{h}) = \sum_{r=1}^{T} \hat{R}_{z_r}(h_r) = \sum_{r=1}^{T} \sum_{i=1}^{m} l(h_r(x_i^r), y_i^r). \tag{3.5}$$

A similar result to that of Bias Learning is given for Multi-Task Learning (Baxter, 2000, Theorem 4):

$$\hat{R}_{\boldsymbol{z}}(\boldsymbol{h}) \leq R_{\boldsymbol{P}}(\boldsymbol{h}) + \epsilon,$$

with probability $1 - \eta$ given that the number of samples per task

$$m \geq \max\left(\frac{64}{T\epsilon^2} \log \frac{4C\left(\frac{\epsilon}{16}, \mathbb{H}^T\right)}{\eta}, \frac{16}{\epsilon^2} \right).$$

Observe that we do not need the *distribution-driven* capacity in this case, just the *sample-driven* capacity.

Feature Learning

Feature Learning is a common way to encode bias. The most popular example are Neural Networks, where all the hidden layers can be seen as a Feature Learning engine that learns a mapping from the original space to a space with "strong" features. In general, a set of "strong" feature maps is defined as $\mathcal{F} = \{f, f : \mathcal{X} \to \mathcal{V}\}$. Using these features, functions $g \in \mathcal{G}$ (which are tipically simple) are built: $\mathcal{X} \to_f \mathcal{V} \to_g \mathcal{Y}$. Thus, for each map f, the hypotheses space can be expressed as $\mathcal{H}_f = \{h = \mathcal{G} \circ f, g \in \mathcal{G}\}$, and the family of hypotheses spaces is $\mathbb{H} = \{\mathcal{H}_f, f \in \mathcal{F}\}$. Now, the Bias Learning problem is the problem of finding a good mapping f. It is proved (Baxter, 2000, Theorem 6) that in the Feature Learning case the capacities of $\mathbb{H}$ can be bounded by the capacities of $\mathcal{F}$ and $\mathcal{G}$ as

$$\begin{aligned} C\left(\epsilon, \mathbb{H}^T\right) &\leq C\left(\epsilon_1, \mathcal{G}\right)^T C_{\mathcal{G}_\ell}(\epsilon_2, \mathcal{F}), \\ C\left(\epsilon, \mathbb{H}^*\right) &\leq C_{\mathcal{G}_\ell}(\epsilon, \mathcal{F}) \end{aligned}$$

with $\epsilon = \epsilon_1 + \epsilon_2$. Here, $C_{\mathcal{G}_\ell}(\epsilon, \mathcal{F})$ is defined as $C_{\mathcal{G}_\ell}(\epsilon, \mathcal{F}) := \sup_P \mathcal{N}(\epsilon, \mathcal{F}, d_{[P, \mathcal{G}_\ell]})$, where

$$d_{[P, \mathcal{G}_\ell]}(f, f') = \int_{\mathcal{X} \times \mathcal{Y}} sup_{g \in \mathcal{G}} \left| \ell\left(g \circ f(x), y\right) - \ell\left(g \circ f'(x), y\right) \right| dP(x, y)$$

is a pseudo-metric. Using these results alongside those presented for Bias Learning is useful to establish bounds for Feature Learning models like Neural Networks.

Generalized VC-Dimension for Multi-Task Learning

The concepts presented until now rely on the concepts of two capacities of a family of hypotheses spaces $\mathbb{H}$ to establish bounds in the difference $\hat{R}_{\boldsymbol{z}}(\boldsymbol{h}) - R_Q(\boldsymbol{h})$, that is, the

probability of deviations between the empirical and expected risks for a given hypothesis sequence. However, it would be more useful to find some result concerning the empirical error and the *best expected error*. To achieve this, a generalized VC-dimension is developed in Baxter (2000) for Multi-Task Learning with Boolean hypothesis.

Definition 3.4. Let $\mathcal{H}$ be a space of boolean functions and $\mathbb{H}$ a boolean hypotheses space family. Denote the set of $T \times m$ matrices in $\mathcal{X}$ as $\mathcal{X}^{T\times m}$ For each $X \in \mathcal{X}^{T\times m}$ and each $\mathcal{H} \in \mathbb{H}$ define the set of binary $T \times m$ matrices

$$\mathcal{H}_{|X} := \left\{ \begin{pmatrix} h\left(x_1^1\right) & \dots & h\left(x_m^1\right) \\ \vdots & \ddots & \vdots \\ h\left(x_1^T\right) & \dots & h\left(x_m^T\right) \end{pmatrix}, h \in \mathcal{H} \right\},$$

and the corresponding family of such sets as

$$\mathbb{H}_{|X} = \bigcup_{\mathcal{H}\in\mathbb{H}} \mathcal{H}_{|X}.$$

For each $T, m \geq 0$ define the number of binary matrices obtainable with $\mathbb{H}$ as

$$\Pi_{\mathbb{H}}(T, m) := \max_{X\in\mathcal{X}^{T\times m}} \left|\mathbb{H}_{|X}\right|.$$

Note that $\Pi_{\mathbb{H}}(T, m) \leq 2^{Tm}$ and if $\Pi_{\mathbb{H}}(T, m) = 2^{Tm}$ we say that $\mathbb{H}$ shatters $\mathcal{X}^{T\times m}$. For each $T > 0$ define

$$d_{\mathbb{H}}(T) := \max_{m:\Pi_{\mathbb{H}}(T,m)=2^{Tm}} m,$$

Here, $d_{\mathbb{H}}(T)$ is the generalized VC-dimension. Also define

$$\overline{d}(\mathbb{H}) := \mathrm{VCdim}\left(\mathbb{H}^1\right) = \mathrm{VCdim}\left(\bigcup_{\mathcal{H}\in\mathbb{H}} \mathcal{H}\right),$$
$$\underline{d}(\mathbb{H}) := \max_{\mathcal{H}\in\mathbb{H}} \mathrm{VCdim}\left(\mathcal{H}\right).$$

$$d_{\mathbb{H}}(T) \geq \max\left(\left\lfloor \frac{\overline{d}(\mathbb{H})}{T} \right\rfloor, \underline{d}(\mathbb{H})\right).$$

where it can be observed that

$$\overline{d}(\mathbb{H}) \geq d_{\mathbb{H}}(T) \geq \underline{d}(\mathbb{H}). \tag{3.6}$$

Now we can present the relevant result expressed in (Baxter, 2000, Corollary 13).

Theorem 3.5. *Given a sequence* $\boldsymbol{P} = (P_1, \dots, P_T)$ *on* $(\mathcal{X} \times \{0, 1\})^T$*, and a sample* $\boldsymbol{z}$ *from this distribution. Consider also a sequence* $\boldsymbol{h} = (h_1, \dots, h_T)$ *of boolean hypothesis* $h_i \in \mathcal{H}$*, then for every* $\epsilon > 0$

$$\left|R_{\boldsymbol{P}}(\boldsymbol{h}) - \hat{R}_{\boldsymbol{z}}(\boldsymbol{h})\right| \leq \epsilon,$$

with probability $1-\eta$ *given that the number of samples per task*

$$m \geq \frac{88}{\epsilon^2}\left[2d_{\mathbb{H}}(T)\log\frac{22}{\epsilon}+\frac{1}{T}\log\frac{4}{\eta}\right]. \tag{3.7}$$

Here, since $d_{\mathbb{H}}(T) \geq d_{\mathbb{H}}(T+1)$, it is easy to see that as the number of task T increases, the number of examples needed per task can decrease. Moreover, as shown in (Baxter, 2000, Theorem 14), if this bound on m is not fulfilled, then we can always find a sequence of distributions $\boldsymbol{P}$ such that

$$\inf_{h\in\mathcal{H}} \hat{R}_{\boldsymbol{z}}(\boldsymbol{h}) > \inf_{h\in\mathcal{H}} R_{\boldsymbol{P}}(\boldsymbol{h}) + \epsilon.$$

With this results we can see that the condition (3.7) has some important properties:

- It is a computable bound, given that we know how to compute $d_{\mathbb{H}}(T)$.
- It provides a sufficient condition for the uniform convergence (in probability) of the empirical risk to the expected risk.
- It provides a necessary condition for the consistency of Multi-Task Learners, i.e. uniform convergence of the best empirical risk to the best expected risk.

3.2.2 Learning with Related Tasks

Using the work of Baxter (2000) as the foundation, several important notions and results are presented in Ben-David and Borbely (2008) for boolean hypothesis functions defined over $\mathcal{X}\times\{0,1\}$. One of the main contributions of this work is a notion of task relatedness. In Baxter (2000) the tasks are related by sharing a common inductive bias that can be learned. In Ben-David and Borbely (2008) a precise mathematical definition for task relatedness is given. The other important contribution is the focus on the individual risk of each task. In Baxter (2000) all the results are given for the Multi-Task empirical and expected risks, which are an average of the risks of each task. However, bounding this average does not establish a sharp bound of the risk of each particular task. This is specially relevant if we are in a Transfer Learning scenario, where there is a target task that we want to solve and the remaining tasks can be seen as an aid to improve the performance in the target.

A Notion of Task Relatedness: $\mathcal{F}$-Related Tasks

The main concept for the theory developed in Ben-David and Borbely (2008) is a set of $\mathcal{F}$ of transformations $f:\mathcal{X}\to\mathcal{X}$. Given a probability distribution F over $\mathcal{X}\times\{0,1\}$, a set of tasks with distributions $P_1,\ldots,P_T$ are $\mathcal{F}$-related if, for each task there exists some $f_i\in\mathcal{F}$ such that $P_i = f_i(F)$.

Definition 3.6 ($\mathcal{F}$-related task)**.** Consider a measurable space $(\mathcal{X},\mathcal{A})$ and the corresponding measurable product space $(\mathcal{X}\times\{0,1\},\mathcal{A}\times\wp(\{0,1\}))$, where $\wp(\Omega)$ is the powerset of set Ω. Consider P a probability distribution over this product space and a function $f:\mathcal{X}\to\mathcal{X}$, then we define the distribution $f[P]$ such that for any $S\in\mathcal{A}$,

$$f[P](S) := P(\{(f(x),b),(x,b)\in S\}).$$

Let $\mathcal{F}$ be a set of transformations $f : \mathcal{X} \to \mathcal{X}$, and let P_1, P_2 be distributions over $(\mathcal{X} \times \{0,1\}, \mathcal{A} \times \wp(\{0,1\}))$, then the distributions P_1, P_2 are $\mathcal{F}$-related if $f[P_1] = P_2$ or $f[P_2] = P_1$ for some $f \in \mathcal{F}$.

This notion establishes a clear definition of related tasks but we are interested in how a learner can use this relatedness to improve the learning process. For that, considering that $\mathcal{F}$ is a group under function composition, we regard at the action of the group $\mathcal{F}$ over the set of hypotheses $\mathcal{H}$. This action defines the following equivalence relation in $\mathcal{H}$:

$$h_1 \sim_{\mathcal{F}} h_2 \iff \exists f \in \mathcal{F}, h_1 \circ f = h_2.$$

This equivalence relation defines equivalence classes $[h]$, that is let $h' \in \mathcal{H}$ be an hypothesis, then $h' \in [h]$ iff $h' \sim_{\mathcal{F}} h$. We consider the quotient space

$$\mathcal{H}_{\mathcal{F}} := \mathcal{H}/\sim_{\mathcal{F}} = \{[h], h \in \mathcal{H}\}.$$

It is important to observe that $\mathcal{H}_{\mathcal{F}} = \mathbb{H}'$ is a hypotheses space family, since it is a set of equivalence classes $[h] = \mathcal{H}'$, which are sets of hypotheses.

The Multi-Task Empirical Risk Minimization

This equivalence classes are useful to divide the learning process in two stages, this is called the *Multi-Task ERM*. Consider the samples $z_1, \ldots, z_T$ from T different tasks, then

1. Select the best hypothesis class $[h^{\mathcal{F}}] \in \mathcal{H}_{\mathcal{F}}$:

$$[h^{\mathcal{F}}] := \min_{[h] \in \mathcal{H}_{\mathcal{F}}} \inf_{h_1, \ldots, h_T \in [h]} \frac{1}{T} \sum_{r=1}^{T} \hat{R}_{z_i}(h_i),$$

2. Select the best hypothesis $h^{\diamond}$ for the target task (without loss of generality, consider the first one):

$$h^{\diamond} = \inf_{h \in [h^{\mathcal{F}}]} \hat{R}_{z_1}(h).$$

For example, consider the handwritten digits recognition problem, we might integrate T different datasets designed in different conditions. Each dataset have been created using certain conditions of light and some specific scanner for getting the images. Even different pens or pencils might be influential in the stroke of the numbers. All these conditions are the $\mathcal{F}$ transformations, and each $f \in \mathcal{F}$ generate a different bias for the dataset. However, there exists a probability for "pure" digits, e.g. the pixels of digit one have higher probability around a line in the middle of the picture than in the sides. This "pure" probability distribution P_0 and all the distributions $P_1, \ldots, P_T$ from which our datasets have been sampled might be $\mathcal{F}$-related among them and with P_0. If we first determine the $\mathcal{F}$-equivalent class of hypothesis $[h]$ suited for digit recognition in the first stage, then it will be easier to select $h_1, \ldots, h_T \in [h]$ for each dataset in the second one.

Bounds for $\mathcal{F}$-Related Tasks

The results of Theorem 3.5 can be applied to the hypothesis quotient space of equivalent classes $\mathcal{H}_{\mathcal{F}}$. However the following results is needed first. Let P_1, P_2 be $\mathcal{F}$-related distributions, then this statement can be proved (Ben-David and Borbely, 2008, Lemma 2):

$$\inf_{h\in\mathcal{H}} R_{P_1}(h) = \inf_{h\in\mathcal{H}} R_{P_2}(h). \tag{3.8}$$

This indicates that the the expected risk is invariant under transformations of $\mathcal{F}$. Now, one of the main results (Baxter, 2000, Theorem 2) can be given.

Theorem 3.7. *Let $\mathcal{F}$ be a set of transformations $f : \mathcal{X} \to \mathcal{X}$ that is a group under function composition. Let $\mathcal{H}$ be a hypotheses space so that $\mathcal{F}$ acts as a group over $\mathcal{H}$, and consider the quotient space $\mathcal{H}_{\mathcal{F}} = \{[h], h \in \mathcal{H}\}$. Consider $\boldsymbol{P} = (P_1, \dots, P_T)$ a sequence of $\mathcal{F}$-related distributions over $\mathcal{X} \times \{0, 1\}$, and $\boldsymbol{z} = (z_1, \dots, z_T)$ the corresponding sequence of samples where z_i is sampled using P_i, then for every $[h] \in \mathcal{H}_{\mathcal{F}}$ and $\epsilon > 0$*

$$\left| \inf_{h_1,\dots,h_T\in[h]} \frac{1}{T}\sum_{r=1}^{T} \hat{R}_{z_r}(h_r) - \inf_{h'\in[h]} R_{P_1}(h') \right| \leq \epsilon$$

with probability greater than η if the number of samples from each distribution satisfies

$$|z_i| \geq \frac{88}{\epsilon^2}\left[2d_{\mathcal{H}_{\mathcal{F}}}(T)\log\frac{22}{\epsilon} + \frac{1}{T}\log\frac{4}{\eta}\right]. \tag{3.9}$$

Note that, in contrast to Theorem 3.5, this result bounds the expected risk of a single task, not the average risk. This is the consequence of applying Theorem 3.5 and substituting the average empirical error using the result from (3.8). Also observe that here the hypotheses space family used is the quotient space $\mathcal{H}_{\mathcal{F}}$, and the VC-dimension of such family is used. Using this result, a bound for learners using the Multi-Task ERM principle is given (Ben-David and Borbely, 2008, Theorem 3)

Theorem 3.8. *Consider $\mathcal{F}$ and $\mathcal{H}$ as in the previous theorem. Consider also the previous sequences of distributions $(P_1, \dots, P_T)$ and corresponding samples $(z_1, \dots, z_T)$. Consider $\underline{d}(\mathcal{H}_{\mathcal{F}}) = \max_{h\in\mathcal{H}} VCdim([h])$. Let $h^\diamond$ be the hypothesis selected using the Multi-Task ERM principle, then for every $\epsilon_1, \epsilon_2 > 0$*

$$\hat{R}_{z_1}(h^\diamond) - \inf_{h'\in\mathcal{H}} R_{P_1}(h') \leq 2(\epsilon_1 + \epsilon_2)$$

with probability greater than η if

$$|z_1| \geq \frac{64}{\epsilon^2}\left[2\underline{d}(\mathcal{H}_{\mathcal{F}})\log\frac{12}{\epsilon} + \frac{1}{T}\log\frac{8}{\eta}\right], \tag{3.10}$$

and for $i \neq 1$

$$|z_i| \geq \frac{88}{\epsilon^2}\left[2d_{\mathcal{H}_{\mathcal{F}}}(T)\log\frac{22}{\epsilon} + \frac{1}{T}\log\frac{8}{\eta}\right]. \tag{3.11}$$

The idea of the proof of this theorem helps to understand how using different tasks can help to improve the performance in the target task. Consider $h^* = \inf_{h\in\mathcal{H}} R_{P_1}(h)$ the best hypothesis for the P_1 distribution. According to Theorem 3.7, for $[h^*]$ we have

that

$$\inf_{h_1,\dots,h_T\in[h^*]} \frac{1}{T}\sum_{r=1}^{T} \hat{R}_{z_r}(h_r) \leq \inf_{h'\in[h^*]} R_{P_1}(h') + \epsilon_1.$$

Also, in the first stage of Multi-Task ERM principle, we select the hypothesis class $[h^{\mathcal{F}}]$ that minimizes $\inf_{\boldsymbol{h}\in[h]} R_{\boldsymbol{P}}(\boldsymbol{h})$ where $\boldsymbol{h}$ is a sequence of hypothesis of $\mathcal{H}_{\mathcal{F}}$. According to Theorem 3.7, for $[h^{\mathcal{F}}]$ we have that

$$\inf_{h'\in[h^{\mathcal{F}}]} R_{P_1}(h') \leq \inf_{h_1,\dots,h_T\in[h^{\mathcal{F}}]} \frac{1}{T}\sum_{r=1}^{T} \hat{R}_{z_r}(h_r) + \epsilon_1.$$

Using these two inequalities we get

$$\inf_{h'\in[h^{\mathcal{F}}]} R_{P_1}(h') \leq \inf_{h'\in[h^*]} R_{P_1}(h') + 2\epsilon_1$$

under the condition (3.9). This bounds the risk of the hypotheses space given by the equivalence class of $h^{\mathcal{F}}$ and establishes the inequality (3.11).

Once we select $[h^{\mathcal{F}}]$, the second stage is just ERM using this hypotheses space. According to the Vapnik (1982),

$$\inf_{h\in\mathcal{H}} R_{z_1}(h) - \inf_{h\in\mathcal{H}} R_{P_1}(h) \leq \epsilon_2$$

if

$$|z_1| \geq \frac{64}{\epsilon^2}\left[2\text{VCdim}\left(\mathcal{H}\right)\log\frac{12}{\epsilon} + \frac{1}{T}\log\frac{8}{\eta}\right].$$

Since the ERM will not use the whole space $\mathcal{H}$ but the subset $[h^{\mathcal{F}}] \subset \mathcal{H}$, and

$$\text{VCdim}\left([h^{\mathcal{F}}]\right) \leq \max_{[h]\in\mathcal{H}_{\mathcal{F}}} \text{VCdim}\left([h]\right) = \underline{d}(\mathcal{H}_{\mathcal{F}}).$$

then we can write the inequality (3.10) of the theorem. The advantage of using multiple tasks is then illustrated in this bound and it will be defined by the gap between $\text{VCdim}\left(\mathcal{H}\right)$ and $\underline{d}(\mathcal{H}_{\mathcal{F}})$. If $\underline{d}(\mathcal{H}_{\mathcal{F}})$ is smaller than $\text{VCdim}\left(\mathcal{H}\right)$, the number of samples needed to solve the target task will also be smaller. Also, the sample complexity of the rest of tasks is given by $d_{\mathcal{H}_{\mathcal{F}}}(T)$.

That is, Multi-Task Learning allows to select a subset of hypotheses from which a learner can use the ERM principle. In this stage, the sample complexity is controlled by the generalized VC-dimension of the set of equivalent classes of hypothesis. Once the best equivalent class has been selected, the VC-dimension of this subset, compared to the VC-dimension of the whole set of hypotheses, is what marks the difference between Single Task and Multi-Task Learning.

Analysis of generalized VC-dimension with $\mathcal{F}$-related tasks

As we have seen in Theorem 3.8, the VC-dimensions $\text{VCdim}\left(\mathcal{H}\right), \underline{d}(\mathcal{H}_{\mathcal{F}})$ and $d_{\mathcal{H}_{\mathcal{F}}}(T)$ are crucial for stating the advantage of Multi-Task over Single Task Learning. To understand better how these concepts interact, Ben-David et al. give some theoretical results. Recall that, given a hypotheses space $\mathcal{H}$, $\mathcal{H}_{\mathcal{F}}$ is a family of hypotheses spaces composed by the

hypotheses spaces $[h], h \in \mathcal{H}$, then

$$
\begin{aligned}
d_{\mathcal{H}_\mathcal{F}}(T) &= \max_{\left\{m, \Pi_{\mathcal{H}_\mathcal{F}} = 2^{Tm}\right\}} m, \\
\underline{d}(\mathcal{H}_\mathcal{F}) &= max_{h \in \mathcal{H}} \mathrm{VCdim}\left([h]\right), \\
\overline{d}(\mathcal{H}_\mathcal{F}) &= \mathrm{VCdim}\left(\bigcup_{[h] \in \mathcal{H}_\mathcal{F}} [h]\right) = \mathrm{VCdim}\left(\mathcal{H}\right).
\end{aligned}
$$

Using the result from (3.6) we observe that

$$
\underline{d}(\mathcal{H}_\mathcal{F}) \leq d_{\mathcal{H}_\mathcal{F}}(T) \leq \mathrm{VCdim}\left(\mathcal{H}\right).
$$

That is, the best we can hope when bounding the sample complexity in Theorem 3.8 is $\underline{d}(\mathcal{H}_\mathcal{F}) = d_{\mathcal{H}_\mathcal{F}}(T)$. Ben-David et al. give evidence that, with some restrictions on $\mathcal{H}$, this lower bound can be achieved (Ben-David and Borbely, 2008, Theorem 4).

Theorem 3.9. *If the support of h is bounded, i.e. $|\{x \in \mathcal{X}, h(x) = 1\}| < M$, for all $h \in \mathcal{H}$, then there exists T_0 such that for all $T > T_0$*

$$
d_{\mathcal{H}_\mathcal{F}}(T) = \underline{d}(\mathcal{H}_\mathcal{F}).
$$

Thus, a sufficient condition on the hypotheses space $\mathcal{H}$ to achieve the lowest $d_{\mathcal{H}_\mathcal{F}}(T)$ is a bounded support of any hypothesis. Although this condition may be too restricting, it can also be proved that the upper limit of $d_{\mathcal{H}_\mathcal{F}}(T)$, that is, $\mathrm{VCdim}\left(\mathcal{H}\right)$, under some conditions on $\mathcal{F}$ is not achieved.

The following result (Ben-David and Borbely, 2008, Theorem 6) shows this.

Theorem 3.10. *If $\mathcal{F}$ is finite and $\frac{T}{\log(T)} \geq VCdim\left(\mathcal{H}\right)$, then*

$$
d_{\mathcal{H}_\mathcal{F}}(T) \leq 2\log(|\mathcal{F}|)
$$

This inequality indicates that, given a finite set of transformation $\mathcal{F}$, there are scenarios when $\mathrm{VCdim}\left(\mathcal{H}\right)$ is arbitrarily large but $d_{\mathcal{H}_\mathcal{F}}(T)$ is bounded, and therefore, the right-hand side of inequality (3.11) is also bounded. That is, the Multi-Task bound, which substitutes $\mathrm{VCdim}\left(\mathcal{H}\right)$ by $d_{\mathcal{H}_\mathcal{F}}(T)$ is a better one in this cases.

3.2.3 Other bounds for Multi-Task Learning

The work of Baxter Baxter (2000) set the foundations for the theoretical analysis of MTL and LTL. In this work, an MTL extension to the VC-dimension is given, and it is used to develop some results bounding the difference between the Multi-Task empirical and expected risks

$$
\left| R_{\boldsymbol{P}}(\boldsymbol{h}) - \hat{R}_{\boldsymbol{z}}(\boldsymbol{h}) \right|
$$

for any sequence of hypothesis $\boldsymbol{h} \in \mathcal{H}^T$, see Theorem 3.5. This is a necessary condition for the consistency of Multi-Task Learners, but not a sufficient one. Then, Ben-David et al. Ben-David and Borbely (2008); Ben-David and Schuller (2003) defines a notion of task relatedness, see Definition 3.6. Using this notion and building an appropiate hypotheses space $\mathcal{H} = [h]$, they bound the *excess risk* of an Empirical Multi-Task Learner, that is, the difference between the best empirical risk, achieved by such Learner, and the best

possible expected risk (Theorem 3.8)

$$\left| \inf_{\boldsymbol{h} \in \mathcal{H}^T} R_{\boldsymbol{P}}(\boldsymbol{h}) - \inf_{\boldsymbol{h} \in \mathcal{H}^T} \hat{R}_{\boldsymbol{z}}(\boldsymbol{h}) \right|.$$

Moreover, using this task relatedness definition not only the Multi-Task average excess risk is bounded, but the individual excess risk of each task (Theorem 3.10).

The works discussed until this point on the VC-dimension, and the corresponding extensions to the MTL framework expressed in Definition 3.4, to bound the differences between empirical and expected risks. However in Ando and Zhang (2005), the authors rely on another notion of complexity, the Rademacher Complexity Bartlett and Mendelson (2002), which measures how well a family of hypothesis can approximate random noise. The Rademacher complexity, unlike the VC-dimension, is distribution-dependent. That is the VC-dimension only uses properties of the hypotheses space $\mathcal{H}$, while the Rademacher complexity also depends on the data distribution F. Other theoretical works obtain bounds for linear feature extractors methods for MTL, such as Cavallanti et al. (2010); Maurer (2006a,b). In the more general case of LTL, some improved bounds are found for specific cases like the trace norm regularized MTL models Maurer et al. (2013). Then, in Maurer et al. (2016) bounds for a wide class of MTL models based on Multi-Task Representation Learning (MTRL) are given, in both an MTL and an LTL setting. This bounds are not dependent on the data dimensions, as other bounds for linear models, and use an approach based on empirical process theory instead of the generalized VC-dimension to derive these bounds.

3.2.4 Learning Under Privileged Information

Another important motivation for Multi-Task Learning can be found in the Learning Under Privileged Information paradigm Vapnik and Izmailov (2015). The standard machine learning paradigm tries to find the hypothesis h from a set of hypotheses $\mathcal{H}$ that minimizes the expected risk $\hat{R}_z$ given a set of training samples. Vapnik is one of the main contributors to the theory of statistical learning Vapnik (2000). In this theory several important results are provided: necessary and sufficient conditions for the consistency of learning processes and bounds for the rate of convergence, which uses the notion of VC-dimension. A new inductive principle, Structural Risk Minimization (SRM), and an algorithm, Support Vector Machine (SVM), that makes use of this notion to improve the learning process.

Nowadays learning approaches based on Deep Neural Networks, which are not focused on controlling the capacity of the set of hypotheses, outperform the SVM approaches in many problems. However, these popular Deep Learning approaches require large amounts of data to learn good hypothesis. It is commonly believed that machines need much more samples to learn than humans do. Vapnik Vapnik and Izmailov (2015); Vapnik and Vashist (2009) reflects on this belief and states that humans typically learn under the supervision of an Intelligent Teacher. This Teacher shares important knowledge by providing metaphors, examples or clarifications that are helpful for the students.

LUPI Paradigm

The additional knowledge provided by the Teacher is the Privileged Information that is available only during the training stage. To incorporate the concept of Intelligent Teacher in the Machine Learning framework, Vapnik introduces the paradigm of Learning Under

Privileged Information (LUPI). In the LUPI paradigm describes the following model. Given a set of i.i.d. triplets

$$z = \{(x_1, x_1^*, y_1), \ldots, (x_n, x_n^*, y_n)\}, \; x \in \mathcal{X}, x^* \in \mathcal{X}^*, y \in \mathcal{Y}$$

generated according to an unknown distribution $P(x, x^*, y)$, the goal is to find the hypothesis $h(x, \alpha^*)$ from a set of hypotheses $\mathcal{H} = \{h(x, \alpha), \alpha \in A\}$ that minimizes some expected risk

$$R_F = \int \ell(h(x, \alpha), y) \, dF(x, y).$$

Note that the goal is the same that in the standard paradigm, however with the LUPI approach we are provided additional information, which is available only during the training stage. This additional information is encoded in the elements x^* of a space $\mathcal{X}^*$, which is different from $\mathcal{X}$. The goal of the Teacher is, given a pair (x_i, y_i), to provide a useful information $x^* \in \mathcal{X}^*$ given some probability $P(x^* \mid x)$. That is, the "intelligence" of the Teacher is defined by the choice of the space $\mathcal{X}^*$ and the conditional probability $P(x^* \mid x)$. To understand better this paradigm consider the following example.

Example. Consider that the goal is to find a decision rule that classifies biopsy images into cancer or non-cancer. Here, $\mathcal{X}$ is the space of images, i.e. the matrix of pixels, for example $[0, 1]^{64 \times 64}$. The label space is $\mathcal{Y} = \{0, 1\}$. An Intelligent Teacher might provide a student of medicine with commentaries about the images, for example: "There is an area of unusual concentration of cells of Type A." or "There is an aggresive proliferation of cells of Type B". These commentaries are the elements x^* of certain space X^* and the Teacher also chooses the probability $P(x^* \mid x)$.

Analysis of convergence rates

To get a better insight of how the Privileged Information can help in the Learning process, Vapnik provides a theoretical analysis of its influence on the learning rates. In the standard learning paradigm, how well the expected risk R_F can be bounded is controlled by two factors: the empirical risk $\hat{R}_z$ and the VC-dimension of the set of hypotheses $\mathcal{H}$. In the case of classification, where $\mathcal{Y} = \{-1, 1\}$ and the loss $\ell(h(x, \alpha), y) = \mathbf{1}_{h(x,\alpha)y \leq 0}$, the risks can be expressed as

$$R_F(\alpha) = \int \mathbf{1}_{yh(x,\alpha) \leq 0} dF(x, y) = P(h(x, \alpha) \, y \leq 0),$$
$$\hat{R}_z(h(\cdot, \alpha)) = \sum_{i=1}^{n} \mathbf{1}_{y_i h(x_i, \alpha) \leq 0} = \nu(\alpha).$$

In (Vapnik, 1982, Theorem 6.8) the following bound for the rate of convergence is given with probability $1 - \eta$:

$$P(h(x, \alpha_n) \, y \leq 0) \leq \nu(\alpha_n) + O\left(\frac{d \log\left(\frac{2n}{d}\right) - \log \eta}{n} \sqrt{\nu(\alpha_n) \frac{n}{d \log\left(\frac{2n}{d}\right) - \log \eta}} \right).$$

That is, the bound is controlled by the ratio d/n, where d is the $\mathrm{VCdim}(\mathcal{H})$. If this VC-dimension is finite, the bound goes to zero as n grows. However, two different cases can be considered.

Separable case: the training data can be classified in two groups without errors. That is, there exists $\alpha_n \in \eta$ such that $y_i h\left(x_i, \alpha_n\right) > 0$ for $i = 1, \ldots, n$, and thus $\nu(\alpha_n) = 0$. In this case, the following bound for the rate of converge holds

$$P(h\left(x, \alpha_n\right) y \leq 0) \leq O\left(\frac{d \log\left(\frac{2n}{d}\right) - \log \eta}{n}\right).$$

Non-Separable case: the training data cannot be classified in two groups without errors. That is, for all $\alpha_n \in A$, there exists $i = 1, \ldots, n$, such that $y_i h\left(x_i, \alpha_n\right) \leq 0$, and thus $\nu(\alpha_n) > 0$. In this case, the following bound for the rate of converge holds

$$P(h\left(x, \alpha_n\right) y \leq 0) \leq \nu(\alpha_n) + O\left(\sqrt{\frac{d \log\left(\frac{2n}{d}\right) - \log \eta}{n}}\right).$$

Note that there is an important difference here in the rate of convergence. The separable case has a convergence rate of d/n, while the non-separable case has a rate of $\sqrt{d/n}$. Vapnik tries to address the question of why there exists such difference.

Oracle SVM

Vapnik tries to answer these question by looking at Support Vector Machines. In the separable case, one has to minimize the functional

$$J(w) = \|w\|^2$$

subject to the constraints

$$y_i\left(w x_i + b\right) \geq 1.$$

However, in the non-separable case the functional to minimize is

$$J(w, \xi_1, \ldots, \xi_n) = \|w\|^2 + C \sum_{i=1}^{n} \xi_i$$

subject to the constraints

$$y_i\left(w x_i + b\right) \geq 1 - \xi_i.$$

That is, in the separable case d parameters (of w) have to be estimated using n examples, while in the non-separable case $d + n$ parameters (considering w and the slack variables $\xi_1, \ldots, \xi_n$) have to be estimated with n examples.

What would happen if the parameters $\xi_1, \ldots, \xi_n$ were known? In Vapnik and Izmailov (2015) an *Oracle* SVM is considered. Here, the learner (Student) is supplied with a set of triplets

$$(x_1, \xi_1^0, y_1), \ldots, (x_n, \xi_n^0, y_n)$$

where $\xi_1^0, \ldots, \xi_n^0$ are the slack variables for the best decision rule $h\left(x, \alpha_0\right) = \inf_{\alpha \in A} R_F(h\left(\cdot, \alpha\right))$:

$$\xi_i^0 = \max\left(0, 1 - h\left(x, \alpha_0\right)\right), \ \forall i = 1, \ldots, n.$$

An *Oracle* SVM has to minimize the functional

$$J(w) = \|w\|^2$$

subject to the constraints

$$y_i \left(w x_i + b \right) \geq 1 - \xi_i^0.$$

Since the slack variables ξ_i^0 are known in advance, it can be shown Vapnik and Vashist (2009) that for the *Oracle* SVM the following bound holds

$$P(h\left(x, \alpha_n\right) y \leq 0) \leq P(1 - \xi^0 \leq 0) + O\left(\frac{d \log\left(\frac{2n}{d}\right) - \log \eta}{n} \right),$$

where $P(1 - \xi^0 \leq 0)$ is the probability error of the hypothesis $h\left(x, \alpha_0\right)$. That is, we recover the rate d/n.

From Oracle to Intelligent Teacher

The *Oracle* SVM is a theoretical construct, but we can approximate it by modelling the slack variables with the information provided by the Teacher in the LUPI paradigm. That is, the Teacher defines a space $\mathcal{X}^*$ and a set of functions $\{f^*(x, \alpha^*), \alpha^* \in A^*\}$. Then model the slack variables as

$$\xi^* = f^*(x^*, \alpha^*).$$

From the pairs generated by some random generator in nature, the Teacher also defines the probability $P\left(x^* \mid x\right)$ to provide the triplets

$$(x_1, x_1^*, y_1), \ldots, (x_n, x_n^*, y_n).$$

Then, we can consider the problem where the goal is to minimize

$$J(\alpha, \alpha^*) = \sum_{i=1}^{n} \max(0, f^*(x_i^*, \alpha^*))$$

subject to the constraints

$$h\left(x_i, \alpha\right) \geq 1 - f^*(x_i^*, \alpha^*).$$

Let $f(x, \alpha_n), h(x.\alpha_n)$ that minimize this problem. Then, in (Vapnik and Vashist, 2009, Proposition 2) the following results for the bound of convergence is given

$$P(h\left(x, \alpha_n\right) y \leq 0) \leq P(1 - f^*(x^*, \alpha_n^*) \leq 0) + O\left(\frac{(d + d^*) \log\left(\frac{2n}{(d+d^*)}\right) - \log \eta}{n} \right),$$

where d^* is the VC-dimension of the space of hypothesis $\{f(x, \alpha^*) \in A^*\}$. This result shows that, to maintain the best convergence rate d/n, we need to estimate the $P(1 - f^*(x^*, \alpha_n^*) \leq 0)$. Although this probability is unknown, we can control it. Considering

$$\alpha_0^* = \inf_{\alpha^* \in A^*} \int_{\mathcal{X}^*} \max(0, f^*(x^*, \alpha^*) - 1) dP(x^*)$$

and

$$\alpha_n^* = \inf_{\alpha^* \in A^*} \sum_{i=1}^{n} \max(0, f^*(x_i^*, \alpha^*) - 1).$$

Consider $\{f^*(x^*, \alpha^*), \alpha^* \in A^*\}$ such that $f^*(x^*, \alpha^*) < B, \alpha^* \in A^*$, then

$$\{\max(0, f^*(x^*, \alpha^*) - 1), \alpha^* \in A^*\}$$

is a set of totally bounded non-negative functions, then we have the standard bound Vapnik (2000)

$$P(1 - f^*(x^*, \alpha_0^*) \leq 0) \leq P(1 - f^*(x^*, \alpha_n^*) \leq 0) + O\left(\sqrt{\frac{d^* \log\left(\frac{2n}{d^*}\right) - \log \eta}{n}}\right).$$

with probability $1 - 2\eta$. That is, to have a rate of d/n for α_n, we need to estimate α_n^*, which has a rate of $\sqrt{d^*/n}$. However, observe that $\mathcal{X}^*$ is the space suggested by the Teacher, which hopefully has a much lower capacity, and thus, the convergence will be faster in this space.

SVM+

Vapnik describes an extension of the SVM that embodies the LUPI paradigm Vapnik and Izmailov (2015); Vapnik and Vashist (2009). Given a set of triplets

$$(x_1, x_1^*, y_1), \ldots, (x_n, x_n^*, y_n),$$

the idea is to model the slack variables of the standard SVM using the elements $x^* \in \mathcal{X}^*$ as

$$\xi(x^*, y) = [y(w^* \phi^*(x^*) + b^*)]_+ = \max\left(y(w^* \phi^*(x^*) + b^*), 0\right).$$

The minimization problem is the following:

$$\begin{aligned} \underset{w, w^*, b, b^*}{\arg\min} \quad & C \sum_{i=1}^{n} [y_i(\langle w^*, \phi^*(x_i^*)\rangle + b^*)]_+ + \frac{1}{2}\langle w, w\rangle + \frac{\mu}{2}\langle w^*, w^*\rangle \\ \text{s.t.} \quad & y_i(\langle w, \phi(x_i)\rangle + b) \geq 1 - [y_i(\langle w^*, \phi^*(x_i^*)\rangle + b^*)]_+ . \end{aligned} \tag{3.12}$$

Here ϕ and ϕ^* are two transformations that can be different. However, note that problem (3.12) is not convex due to the positive part term in the objective function. Vapnik et al. propose a relaxation of this problem to obtain a convex one. The idea is to model the slack variables ξ as

$$\xi(x^*, y) = [y(w^* \phi^*(x^*) + b^*)] + \zeta(x^*, y),$$

where $\zeta(x^*, y) \geq 0$. The minimization problem is then

$$\begin{aligned} \underset{w, w^*, b, b^*, \zeta_i}{\arg\min} \quad & C \sum_{i=1}^{n} ([y_i(\langle w^*, \phi^*(x_i^*)\rangle + b^*)] + \zeta_i) + C\Delta \sum_{i=1}^{n} \zeta_i \\ & + \frac{1}{2}\langle w, w\rangle + \frac{\mu}{2}\langle w^*, w^*\rangle \\ \text{s.t.} \quad & y_i(\langle w, \phi(x_i)\rangle + b) \geq 1 - [y_i(\langle w^*, \phi^*(x_i^*)\rangle + b^*) + \zeta_i], \\ & y_i(\langle w^*, \phi^*(x_i^*)\rangle + b^*) + \zeta_i \geq 0, \\ & \zeta_i \geq 0, \\ \text{for} \quad & i = 1, \ldots, n. \end{aligned} \tag{3.13}$$

Problem (3.13) is convex and the corresponding dual problem is

$$\begin{aligned} \underset{\alpha_i,\delta_i}{\arg\min} \quad & \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j k(x_i,x_j) + \frac{1}{2\mu}\sum_{i,j=1}^{n} y_i y_j (\alpha_i-\delta_i)(\alpha_j-\delta_i) k^*(x_i^*,x_j^*) - \sum_{i=1}^{n}\alpha_i \\ \text{s.t.} \quad & 0 \le \delta_i \le C \\ & 0 \le \alpha_i \le C + \delta_i, \\ & \sum_{i=1}^{n}\delta_i y_i = 0, \ \sum_{i=1}^{n}\alpha_i y_i = 0, \\ \text{for} \quad & i = 1,\dots,n. \end{aligned} \tag{3.14}$$

where we use the kernel functions

$$k(x_i,x_j) = \langle \phi(x_i), \phi(x_j) \rangle \,, \ k^*(x_i^*,x_j^*) = \left\langle \phi^*(x_i^*), \phi^*(x_j^*) \right\rangle .$$

We can observe in Problem (3.14) that the LUPI paradigm exerts a similarity control, correcting the similarity in space $\mathcal{X}$ with the similarity in the privileged space $\mathcal{X}^*$. For that reason, $\mathcal{X}$ and $\mathcal{X}^*$ are named Decision Space and Correction Space, respectively.

Connection between SVM+ and MTLSVM

In Liang and Cherkassky (2008) the connection between SVM+ and Multi-Task Learning SVM (MTLSVM) is discussed. The MTLSVM proposed in Liang and Cherkassky (2008) is a Multi-Task Learning model based on the SVM. It solves the primal problem

$$\begin{aligned} \underset{w,b,v_r,b_r,\xi_i^r}{\arg\min} \quad & C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\xi_i^r + \frac{1}{2}\langle w,w\rangle + \sum_{r=1}^{T}\frac{\mu}{2}\langle v_r,v_r\rangle \\ \text{s.t.} \quad & y_i^r(\langle w,\phi(x_i^r)\rangle + b + \langle v_r,\phi_r(x_i^r)\rangle + b_r) \ge 1-\xi_i^r, \\ & \xi_i^r \ge 0, \\ \text{for} \quad & r = 1,\dots,T;\ i = 1,\dots,m_r. \end{aligned} \tag{3.15}$$

Here, a combination of a common model for all tasks

$$\langle w, \phi(x_i)\rangle + b$$

and a task-specific model

$$\langle v_r, \phi_r(x_i^r)\rangle + b_r$$

is used. Here, the common transformation ϕ and the task-independent ones ϕ_r can be different The dual problem corresponding to (3.15) is

$$\begin{aligned} \arg\min_{\alpha_i} \quad & \frac{1}{2}\sum_{r,s=1}^{T}\sum_{i,j=1}^{m_r} y_i^r y_j^s \alpha_i^r \alpha_j^s k(x_i^r, x_j^s) + \frac{1}{2\mu}\sum_{r,s=1}^{T}\sum_{i,j=1}^{m_r} y_i^r y_j^s \alpha_i^r \alpha_j^s \delta_{rs} k_r(x_i^r, x_j^s) \\ & - \sum_{r=1}^{T}\sum_{i=1}^{m_r} \alpha_i^r \\ \text{s.t.} \quad & 0 \leq \alpha_i^r \leq C \\ & \sum_{i=1}^{m_r} \alpha_i^r y_i^r = 0, \\ \text{for} \quad & r = 1, \ldots, T;\ i = 1, \ldots, m_r. \end{aligned} \tag{3.16}$$

In Liang and Cherkassky (2008) some similarities between MTLSVM and SVM+ are pointed out. Problem (3.15) can be regarded as an adaptation of (3.13) to solve MTL problems, where different tasks are incorporated and multiple correcting spaces are defined using the transformations ϕ_r. If we consider the problem (3.15) with a single task, it is a modification of the SVM+ problem (3.13) where the slack variables are modeled as

$$\xi(x, y) = y(w^* \phi^*(x) + b^*).$$

That is, it is a relaxation of the original problem (3.13) where the second constraint to model the positive part of the slack variables disappears. This relaxation gives place to some important differences between both models. Since the auxiliary primal variables ζ_i are no longer required, this is reflected in a simpler dual form (3.16), where only n dual variables have to be estimated, instead of the $2n$ dual variables of (3.14). The Multi-Task element is reflected on (3.16) through the δ_{rs} function, which makes the correction of similarity only possible between elements of the same task.

A major remark can be make about the differences between MTLSVM and SVM+. The results for the improved rate of convergence with an Intelligent Teacher may not be valid with MTLSVM, since we are not modelling the slack variables ξ adequately. It is still a work in progress to study the rate of convergence of MTLSVM and to establish more clear links with SVM+.

3.3 Kernels for Multi-Task Learning

The Multi-Task Learning paradigm can be seen as learning a vector-valued function $f : \mathcal{X} \to \mathbb{R}^T$, where each element of the vector corresponds to a different task.

3.3.1 Vector-Valued Reproducing Kernel Hilbert Spaces

Consider $\mathcal{Y}$ a Hilbert space with inner product $(.,.)$, then we can study the Hilbert spaces of functions with values in $\mathcal{Y}$. The kernels in such spaces are operator-valued functions $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$. We look at these spaces from three different and equivalent perspectives: continuous evaluation functionals, positive-definite kernels and feature maps.

Continuous evaluation functionals. The first approach considered is that of continuous evaluation functionals. That is, given a Hilbert space with continuous evaluation functionals, we can find the reproducing kernel operator. Consider the vector-valued Hilbert space $\mathcal{H}$ of functions defined in $\mathcal{X}$ and values in $\mathcal{Y}$

$$\begin{aligned} \mathcal{H} &\to \mathcal{Y} & &\to \mathbb{R} \\ f &\to f(x) & &\to (y, f(x)) \end{aligned}$$

Consider the functionals $L_{x,y}f = (y, f(x))$, if this functionals are continuous, we can apply Riesz Representation theorem. That is, for every $x \in \mathcal{X}, y \in \mathcal{Y}$ we can find an unique $g_{x,y} \in \mathcal{H}$ such that for all $f \in \mathcal{H}$,

$$L_{x,y}f = (y, f(x)) = \langle g_{x,y}, f \rangle_{\mathcal{H}} . \tag{3.17}$$

We can now give the definition of vector-valued Hilbert space from the point of view of continuous functionals (Micchelli and Pontil, 2005, Definition 2.1)

Definition 3.11 (vector-valued RKHS)**.** We say that $\mathcal{H}$ is a vector-valued RKHS when for any $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, the functional $L_{x,y}f = (y, f(x))$ is continuous.

Note that this is a definition similar to the scalar case but we use the inner product of $\mathcal{Y}$ to construct the scalar-valued functionals $L_{x,y}$. The price we pay is that it is necessary to express the Riesz representation as dependent of the elements $y \in \mathcal{Y}$. To get rid of this dependence for every $x \in \mathcal{X}$ we can define the linear operator

$$\begin{aligned} g_x : \mathcal{Y} &\to \mathcal{H} \\ y &\to g_x y = g_{x,y}. \end{aligned}$$

This operator is well defined because $g_{x,y}$ is unique for every $x \in \mathcal{X}, y \in \mathcal{Y}$ and its linearity is easy to check from the linearity of the inner product $(.,.)$.

Using this results can now define the operator

$$\begin{aligned} K(x, \hat{x}) : \mathcal{Y} &\to \mathcal{Y} \\ y &\to K(x, \hat{x})y = (g_{\hat{x}} y)(x) \end{aligned} \tag{3.18}$$

for every $x, \hat{x} \in \mathcal{X}$. Observe that $K(x, \hat{x})$ is linear since g_x is linear. It is possible then to prove that $K(x, \hat{x})$ is a reproducing kernel in $\mathcal{H}$ as seen (Micchelli and Pontil, 2005, Propositon 2.1). To do that, first we have to define a vector-valued kernel and the corresponding reproducing property.

Definition 3.12 (operator-valued Kernel)**.** If $\mathcal{Y}$ is a finite-dimensional Hilbert space, an operator-valued kernel is a function

$$K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$$

which is symmetric and positive definite.

For the clarity of the text an operator-valued K will be referred just as kernel unless an explicit distinction is needed.

Definition 3.13 (Reproducing Property of operator-valued operators)**.** If $\mathcal{Y}$ is a Hilbert space, a function

$$K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$$

has the reproducing property if $\forall x \in \mathcal{X}, y \in \mathcal{Y}, \forall f \in \mathcal{H}$, $(y, f(x)) = \langle K(\cdot, x)y, f \rangle$.

A kernel with the reproducing property is also called a reproducing kernel. The next proposition shows how we can build a reproducing kernel for a space $\mathcal{H}$ in which the evaluation functionals are continuous.

Proposition 3.14. *If for every $x, \hat{x} \in \mathcal{X}$, the function $K(x, \hat{x})$ is defined as in Equation (3.18), then the function*

$$K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$$

is a reproducing kernel.

Proof. To prove that K is a reproducing kernel we need to see

1. $K(x, \hat{x})$ is bounded for every $x, \hat{x} \in \mathcal{X}$ so K is well defined.
2. K is symmetric: $K(x, \hat{x})^* = K(\hat{x}, x)$.
3. K is positive definite: given $n \in \mathbb{N}$, for any $x_1, \dots, x_n \in \mathcal{X}, y_1, \dots, y_n \in \mathcal{Y}$,

$$\sum_{i,j=1}^{n} (y_i, K(x_i, x_j)y_j) \geq 0.$$

4. It has the reproducing property: $\forall x \in \mathcal{X}, y \in \mathcal{Y}, \forall f \in \mathcal{H}$, $(y, f(x)) = \langle K(\cdot, x)y, f \rangle$.

To prove 1 and 2,observe that by applying (3.17) to $f = g_{\hat{x}}\hat{y}$, for every $x \in \mathcal{X}, y \in \mathcal{Y}$ there exists an unique $g_{x,y} = g_x y$ such that

$$(y, (g_{\hat{x}}\hat{y})(x)) = \langle g_x y, g_{\hat{x}}\hat{y} \rangle . \tag{3.19}$$

and combining this result with (3.18) we get

$$(y, K(x, \hat{x})\hat{y}) = (y, (g_{\hat{x}}\hat{y})(x)) = \langle g_x y, g_{\hat{x}}\hat{y} \rangle .$$

Also, using the definitions,

$$(K(\hat{x}, x)y, \hat{y}) = ((g_x y)(\hat{x}), \hat{y}) = (\hat{y}, (g_x y)(\hat{x})) = \langle g_{\hat{x}}\hat{y}, g_x y \rangle .$$

Since both operators $K(x, \hat{x}), K(\hat{x}, x)$ are linear, by the Uniform Boundness Principle Akhiezer and Glazman (1961), $K(x, \hat{x}), K(\hat{x}, x)$ are bounded (hence continuous) and $K(x, \hat{x})^* = K(\hat{x}, x)$.
To prove 3 we write

$$\sum_{i,j=1}^{n} (y_i, K(x_i, x_j)y_j) = \sum_{i,j=1}^{n} \left\langle g_{x_i} y_i, g_{x_j} y_j \right\rangle = \left\| \sum_{i=1}^{n} g_{x_i} y_i \right\|^2 \geq 0$$

Finally, to prove 4 we use that $\forall x \in \mathcal{X}, y \in \mathcal{Y}, \forall f \in \mathcal{H}, \exists g_{x,y} \in \mathcal{H}$ such that

$$(y, f(x)) = \langle g_{x,y}, f \rangle = \langle g_x y, f \rangle = \langle K(\cdot, x)y, f \rangle .$$

□

Semi-positive definite kernels. The second approach changes the point of view. Given a kernel K, the Hilbert space from which K is the reproducing kernel is built. To do this, we use (Micchelli and Pontil, 2005, Theorem 2.1) which extends the Moore-Aronszanj's Theorem:

Theorem 3.15. *If $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ is a kernel, then there exists a unique (up to an isometry) RKHS which admits K as its reproducing kernel.*

The proof is similar to that of the Moore-Aronszanj's Theorem, considering the space of the completion of the span of $\{K_x = K(\cdot, x), x \in \mathcal{X}\}$.

Feature map. The last approach is based on feature maps, which provide a very simple way of generating kernels.

Lemma 3.16. *Any feature map $\Phi : \mathcal{X} \to \mathcal{L}(\mathcal{W}, \mathcal{Y})$ defines a kernel as*

$$K(x, \hat{x}) = \Phi(x)\Phi(\hat{x})^* : \mathcal{Y} \to \mathcal{Y}. \tag{3.20}$$

Proof. We need to prove that it is bounded, symmetric and positive definite:

1. Since $\Phi(x)$ is continuous, the adjoint is $\Phi(x)^*$ is continuous and the composition $\Phi(x) \circ \Phi(\hat{x})^*$ is also continuous.
2. It is symmetric since $(\Phi(x) \circ \Phi(\hat{x})^*)^* = ((\Phi(x)^*)^* \circ \Phi(\hat{x})^*) = (\Phi(x) \circ \Phi(\hat{x})^*)$.
3. Is positive definite since, given any $x_1, \ldots, x_n \in \mathcal{X}, y_1, \ldots, y_n \in \mathcal{Y}$

$$\begin{aligned}\sum_{i,j=1}^{n} (y_i, K(x_i, x_j)y_j) &= \sum_{i,j=1}^{n} (y_i, \Phi(x_i)\Phi(x_j)^* y_j) \\ &= \sum_{i,j=1}^{n} (\Phi(x_i)^* y_i, \Phi(x_j)^* y_j) = \left\| \sum_{i=1}^{n} \Phi(x_i)^* y_i \right\|^2 \geq 0.\end{aligned}$$

□

Since K as defined in (3.20) is a kernel, according to Theorem 3.15 we can find its corresponding vector-valued hilbert space $\mathcal{H}$.

Representer Theorem for Operator-Valued Kernels

The Representer Theorem is a crucial result in Optimization and Machine Learning. Given a regularized empirical risk, under some assumptions, the theorem gives a precise description of the minimizer f^* as a finite linear combination of functions $K(\cdot, x_i)$ where x_i are part of the empirical sample. This result is extended in (Micchelli and Pontil, 2005, Theorem 4.2) for operator-valued kernels.

Theorem 3.17. *Let $\mathcal{Y}$ be a Hilbert space and let $\mathcal{H}$ be the Hilbert space of $\mathcal{Y}$-valued functions with an operator-valued reproducing kernel K. Let $V : \mathcal{Y}^n \times \mathbb{R}_+ \to \mathbb{R}$ be a function strictly increasing in its second variable and consider the problem of minimizing the functional*

$$E(f) := V((f(x_1), \ldots, f(x_n)), \|f\|^2) \tag{3.21}$$

in $\mathcal{H}$. If f_0 minimizes the E, then $f_0 = \sum_{j=1}^{n} K(\cdot, x_j)c_j$ where $c_j \in \mathcal{Y}$. In addition, if V is strictly convex, the minimizer is unique.

A useful bijection between kernels

The scalar-valued kernels are well known and studied but this is not the case for operator-valued kernels. However, as shown in Baldassarre et al. (2012); Hein et al. (2004) we can find a bijection between operator-valued kernels and scalar-valued ones.

Lemma 3.18. *Let $\mathcal{Y}$ be a finite-dimensional Hilbert space and $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ be an operator-valued kernel. Consider also the scalar-valued kernel $L : (\mathcal{X}, \mathcal{Y}) \times (\mathcal{X}, \mathcal{Y}) \to \mathbb{R}$ such that $L((x, z), (\hat{x}, \hat{z})) = (z, K(x, \hat{x})\hat{z})$, then the map $K \to L$ is a bijection.*

Moreover, if we focus on finite dimensional Hilbert spaces, that is, isomorphic to $\mathbb{R}^d$ given an operator-valued kernel K the corresponding scalar-valued kernel L is defined using the normal basis as

$$L((x, e_r), (\hat{x}, e_s)) = (e_r, K(x, \hat{x})e_s) = K(x, \hat{x})_{rs}.$$

That is, each pair $(x, \hat{x})$ defines a matrix which contains the information of how the different outputs, or tasks, are related.

3.3.2 Tensor Product of Reproducing Kernel Hilbert Spaces

Consider the space of the tensor product of two scalar-valued RKHS' $\mathcal{H}_1 \otimes \mathcal{H}_2$ with reproducing kernels K_1, K_2, where the functions $f \in \mathcal{H}_i$ are defined as $f : \mathcal{X}_i \to \mathcal{Y}_i$ for $i = 1, 2$. This tensor space is also a Hilbert space endowed with the inner product:

$$\begin{array}{lll} \langle , \rangle : (\mathcal{X}_1 \otimes \mathcal{X}_2) \times & (\mathcal{X}_1 \otimes \mathcal{X}_2) \to & \mathbb{R} \otimes \mathbb{R} \\ (f_1 \otimes f_2) & (\hat{f}_1 \otimes \hat{f}_2) \to & \left\langle f_1, \hat{f}_1 \right\rangle \left\langle f_2, \hat{f}_2 \right\rangle . \end{array} \tag{3.22}$$

It is easy to check that this inner product is symmetric because the inner products of both $\mathcal{H}_1$ and $\mathcal{H}_2$ are symmetric. It is linear because the inner products of both $\mathcal{H}_1$ and $\mathcal{H}_2$ are linear and the tensor product is linear. Finally, it is positive definite since $\langle f_1, f_1 \rangle \langle f_2, f_2 \rangle \geq 0$ and $\langle f_1, f_1 \rangle \langle f_2, f_2 \rangle = 0 \implies \langle f_i, f_i \rangle = 0$ for $i = 1$ or $i = 2$; taking $i = 1$ without loss of generality, then $f_1 = 0$, so $f_1 \otimes f_2 = 0$. To apply the Riesz Theorem in this Hilbert space it is necessary to check wether the evaluation functionals are continuous or, equivalently, bounded.

Proposition 3.19 (RKHS as tensor product of RKHS')**.** *The space $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$ with the inner product defined in (3.22) have bounded evaluation functionals $L_{x_1 \otimes x_2}$ defined as $L_{x_1 \otimes x_2}(f_1 \otimes f_2) = L_{x_1}(f_1) \otimes L_{x_2}(f_2) = f_1(x_1) f_2(x_2)$ for $x_1 \otimes x_2 \in \mathcal{X}_1 \otimes \mathcal{X}_2$.*

Proof. It is necessary to ensure that the operators $L_{x_1 \otimes x_2}$ are bounded. Given any $f_1 \otimes f_2 \in \mathcal{H}_1 \otimes \mathcal{H}_2$,

$$\|L_{x_1 \otimes x_2}(f_1 \otimes f_2)\| := \|f_1(x_1) f_2(x_2)\| \leq \|f_1(x_1)\| \, \|f_2(x_2)\| \leq M_{x_1} \|f_1\|_{\mathcal{H}_1} M_{x_2} \|f_2\|_{\mathcal{H}_2} .$$

□

We also need to define the kernel function for this tensor product space.

Proposition 3.20. *The kernel function*

$$\begin{array}{lll} K_1 \otimes K_2 : (\mathcal{H}_1 \otimes \mathcal{H}_2) \times & (\mathcal{H}_1 \otimes \mathcal{H}_2) \to & \mathbb{R} \\ (x_1 \otimes x_2) & (\hat{x}_1 \otimes \hat{x}_2) \to & K_1(x_1, x_2) K_2(x_1, x_2) \end{array}$$

is a reproducing kernel for the Hilbert space $\mathcal{H}_1 \otimes \mathcal{H}_2$.

Proof. First, $K_1 \otimes K_2$ is a kernel, that is, it is symmetric and positive-definite. The proof is very similar to the symmetry and positive definitiness proof of the inner product (3.22). To observe that $K_1 \otimes K_2$ has the reproducing property, we write

$$\langle K_1 \otimes K_2(\cdot, x_1 \otimes x_2), f_1 \otimes f_2 \rangle := \langle K_1(\cdot, x_1), f_1 \rangle \langle K_2(\cdot, f_2), x_2 \rangle = f_1(x_1) f_2(x_2).$$

□

Another useful bijection between kernels

To understand the connection between tensor product RKHS' and vector-valued RKHS' we study a special case of operator-valued kernels. A standard assumption is that the relation among the different outputs is independent of the pair $(x, \hat{x})$:

$$K(x, \hat{x}) = k(x, \hat{x})M$$

where k is a scalar-valued kernel and M is some fixed operator $M \in \mathcal{L}(\mathcal{Y})$. That is, the operator $K(x, \hat{x})$ decouples in two parts: the similarity between x and $\hat{x}$ measured by $k(\cdot, \cdot)$ and the interaction between the different outputs expressed by M. In those cases it is easier to express the operator-valued kernel as the tensor product of two spaces. The following lemma shows how we can express such tensor product kernel.

Lemma 3.21. *Let* $\mathcal{Y}$ *be a finite-dimensional Hilbert space and* $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ *be a separable operator-valued kernel, that is* $K(x, \hat{x}) = k(x, \hat{x})M$ *with* $M \in \mathcal{L}^+(\mathcal{Y})$. *Consider the kernel* $K_\mathcal{X} \otimes K_\mathcal{Y} : (\mathcal{X} \otimes \mathcal{Y}) \times (\mathcal{X} \otimes \mathcal{Y}) \to \mathbb{R}$ *such that* $K_\mathcal{X} \otimes K_\mathcal{Y}((x \otimes z), (\hat{x} \otimes \hat{z})) = K_\mathcal{X}(x, \hat{x}) K_\mathcal{Y}(z, \hat{z})$, *with* $K_\mathcal{Y}(z, \hat{z}) = (z, M\hat{z})$ *then the map* $K \to K_\mathcal{X} \otimes K_\mathcal{Y}$ *is a bijection.*

Proof. First, observe that $K_\mathcal{Y}$ is the reproducing kernel of $\mathcal{Y}$ with the inner product induced by the operator $M^{-1} \in \mathcal{L}^+(\mathcal{Y})$. By the Lemma 3.18 there exists a bijection between K and L where $L((x, z), (\hat{x}, \hat{z})) = (z, K(x, \hat{x})\hat{z})$. When $K(x, \hat{x}) = k(x, \hat{x})M$, we define

$$K_\mathcal{X} \otimes K_\mathcal{Y}(x \otimes z, \hat{x} \otimes \hat{z}) = K_\mathcal{X}(x, \hat{x}) K_\mathcal{Y}(z, \hat{z}) = K_\mathcal{X}(x, \hat{x})\,(z, M\hat{z}) = L((x, z), (\hat{x}, \hat{z})),$$

and the bijection is trivial by definition.

Moreover, observe that using this kernel with a basis of $\mathcal{Y}$, $z = e_r, \hat{z} = e_s$,

$$K_\mathcal{X} \otimes K_\mathcal{Y}((x \otimes e_r), (\hat{x} \otimes e_s)) = K_\mathcal{X}(x, \hat{x})(M)_{rs}.$$

□

This kind of kernels are called separable kernels Álvarez et al. (2012); Kadri et al. (2016), however their connection with operator-valued kernels is not very clear. This subsection tries to explain the construction of separable kernels and how they relate to operator-valued kernels.

3.3.3 Using Kernels in Multi-Task Learning

There exists a plethora of work about Single-Task Learning within regularization theory, some general formulation is

$$\sum_{i=1}^{n} \ell(y_i, \langle w, \phi(x_i) \rangle) + \lambda \langle w, w \rangle . \tag{3.23}$$

Here, ℓ is the loss function and ϕ is a transformation to include non-linearity. Popular models such as Ridge Regression or SVMs are particular cases of this formulation for different choices of ℓ. One crucial result for problems using this formulation is the *Representer Theorem*, which states that the any minimizer of problem (3.23) has the form

$$w = \sum_{i=1}^{n} c_j \phi(x_j). \tag{3.24}$$

Given w represented as in (3.24), we write

$$\langle w, \phi(\hat{x}) \rangle = \sum_{i=1}^{n} c_j \langle \phi(x_j), \phi(\hat{x}) \rangle .$$

This is very useful because we can apply the kernel trick and use the transformations ϕ only implicitly. In this subsection it is shown how a broad class of Multi-Task problems can be expressed as regularized Single-Task problems.

Linear MTL Models

Building upon the ideas discussed in Evgeniou and Pontil (2004), two useful results are presented in Evgeniou et al. (2005), which show how we can apply Single-Task Learning methods to Multi-Task Learning problems. The first result (Evgeniou et al., 2005, Proposition 1) is defined for linear models and illustrates under which conditions we can adapt these results for MTL. Consider the linear MTL problem where we want to estimate the task parameters $u_r : r = 1, \ldots, T$, so we define $\boldsymbol{u}^\intercal = (u_1^\intercal, \ldots, u_T^\intercal) \in \mathbb{R}^{Td}$. Then we want to minimize

$$R(\boldsymbol{u}) = \sum_{r=1}^{T} \sum_{i=1}^{m} \ell(y_i^r, \langle u_r, x_i^r \rangle) + \mu J(\boldsymbol{u}), \tag{3.25}$$

where $J(\boldsymbol{u}) = \boldsymbol{u}^\intercal E \boldsymbol{u}$ is the regularizer where different choices of $J(\boldsymbol{u})$, i.e. choices of the matrix E, we can encode different beliefs about the task structure. For example, if $J(\boldsymbol{u}) = \sum_{r=1}^{T} \|u_r\|^2$ the problem decouples and we get independent task learning, so there is no relation among tasks; if $J(\boldsymbol{u}) = \sum_{r,s=1}^{T} \|u_r - u_s\|^2$ we are enforcing the parameters from different tasks to be close, so we expect all tasks to be similar.

Then, Evgeniou et al. propose to consider a vector $\boldsymbol{w} \in \mathbb{R}^p$ with $p \geq Td$ such that we can express $\langle u_r, x \rangle$ as $\langle B_r^\intercal \boldsymbol{w}, x \rangle$, where B_r is a $p \times d$ matrix yet to be specified. One condition for B_r is to be full rank so we can find such $\boldsymbol{w}$. Note that we can also interpret B_r as a feature map $f : \mathbb{R}^d \to \mathbb{R}^p$ such that $\langle u_r, x \rangle = \langle \boldsymbol{w}, B_r x \rangle$. Observe that using the matrices B_r we have the following MTL kernel

$$\hat{k}(x^r, y^s) = \hat{k}((x, r), (y, s)) = x^\intercal B_r^\intercal B_s y.$$

Using these feature maps we would like to write the MTL problem as a Single-Task problem

$$S(\boldsymbol{w}) = \sum_{r=1}^{T}\sum_{i=1}^{m} \ell(y_i^r, \langle \boldsymbol{w}, B_r x_i^r \rangle) + \mu \langle \boldsymbol{w}, \boldsymbol{w} \rangle , \tag{3.26}$$

We also define the feature matrix B as the concatenation $B = (B_r : r = 1, \dots, T) \in \mathbb{R}^{p \times Td}$, then we present the first result of Evgeniou et al. (2005).

Proposition 3.22. *If the feature matrix B is full rank and we define the matrix E in equation (3.25) as to be $E = (B^\intercal B)^{-1}$ then we have that*

$$S(\boldsymbol{w}) = R(B^\intercal \boldsymbol{w}).$$

and therefore $\boldsymbol{u}^* = B^\intercal \boldsymbol{w}^*$.

One important consequence of this result is that since we can solve the MTL problem (3.25) as the STL problem (3.23), then we can apply the *Representer Theorem*. That is, the solution $\boldsymbol{w}^*$ of problem (3.23) has the form

$$\boldsymbol{w} = \sum_{r=1}^{T}\sum_{i=1}^{m} c_i B_r x_i^r,$$

and the prediction can be expressed as

$$\langle \boldsymbol{w}, \hat{x}^s \rangle = \sum_{r=1}^{T}\sum_{i=1}^{m} \alpha_i^r (x_i^r)^\intercal B_r^\intercal B_s \hat{x}^s = \sum_{r=1}^{T}\sum_{i=1}^{m} c_i \hat{k}(x_i^r, \hat{x}^s).$$

Kernel Extension of MTL Models

Evgeniou et al. also extend this results to kernelized models. In the following lemma (Evgeniou et al., 2005, Lemma 2) the give the conditions under which the extension is possible.

Lemma 3.23. *If G is a kernel on $\mathcal{T} \times \mathcal{T}$ and, for every $r = 1, \dots, T$ there are prescribed mappings $z_r : \mathcal{X} \to \mathcal{T}$ such that*

$$K((x, r), (y, s)) = G(z_r(x), z_s(t)), \ x, t \in \mathcal{X}, \ r, s \in [T] \tag{3.27}$$

then K is a multi-task kernel.

That defines K as a semi-positive functional over the product space $\mathcal{X} \times \mathcal{T}$, which is named multi-task kernel. The mappings described in Evgeniou et al. (2005) are

$$z_r(x) = B_r x$$

where B_r are the $p \times d$ matrices previously defined. Then, two examples of multi-task kernels using this lemma are given. The polynomial kernel is defined as

$$K((x, r), (y, s)) = (x^\intercal B_r^\intercal B_s y)$$

and the multi-task Gaussian kernel is defined as

$$K((x, r), (y, s)) = \exp\left(-\gamma \left\| B_r x - B_s y \right\|^2\right).$$

That is, using the result of Proposition 3.22, when the matrix E has a block structure, we can incorporate the task-regularizer information into the Gaussian kernel using that

$$\begin{aligned}\|B_r x - B_s y\|^2 &= x^\intercal B_r^\intercal B_r x + y^\intercal B_s^\intercal B_s y - 2x_r^\intercal B_r^\intercal B_s y_s \\ &= x^\intercal E_{rr}^{-1} x + y^\intercal E_{ss}^{-1} y - 2x_r^\intercal E_{rs}^{-1} y_s.\end{aligned}$$

That is, we use the task information in the original space and then apply the non-linear transformation.

Alternative Kernel Extension of MTL Models

The kernel extension proposed in Evgeniou et al. (2005) proposes to use a mapping in the original finite space to incorporate the task information and then applying the kernel trick over this new mapped features. However, these results do not permit to perform the, possibly infinite dimensional, mapping corresponding to a kernel and then incorporate the task information in the new space. Here we show another approach which makes it is possible to replicate the results for the infinite-dimensional case by using tensor products. Consider a general Hilbert space $\mathcal{H}$ and the functional

$$R(u_1, \ldots, u_T) = \sum_{r=1}^{T} \sum_{i=1}^{m} \ell(y_i^r, \langle u_r, \phi(x_i^r)\rangle) + \mu \sum_r \sum_s E_{rs} \langle u_r, u_s \rangle , \tag{3.28}$$

where $u_1, \ldots, u_T \in \mathcal{H}$ and E is a $T \times T$ matrix. The following lemma illustrates how to solve this problem as a single task problem.

Lemma 3.24. *The predictions $\langle u_r^*, \phi(x)\rangle$ of the solutions $u_1^*, \ldots, u_T^*$ from the Multi-Task optimization problem (3.28) can be obtained solving the problem*

$$S(\boldsymbol{w}) = \sum_{r=1}^{T} \sum_{i=1}^{m} \ell(y_i^r, \langle \boldsymbol{w}, (B_r \otimes \phi(x_i^r))\rangle) + \mu \boldsymbol{w}^\intercal \boldsymbol{w}, \tag{3.29}$$

where $\boldsymbol{w} \in \mathbb{R}^p \otimes \mathcal{H}$ with $p \geq T$ and B_r are the columns of a full rank matrix $B \in \mathbb{R}^{p \times T}$ such that $E^{-1} = B^\intercal B$.

Proof. Replicating the idea of Evgeniou et al. (2005), we can write

$$\boldsymbol{u} = \sum_{t=1}^{T} e_r \otimes u_r,$$

such that $\boldsymbol{u} \in \mathbb{R}^T \otimes \mathcal{H}$. Then, we can reformulate (3.28) as

$$R(\boldsymbol{u}) = \sum_{r=1}^{T} \sum_{i=1}^{m} \ell(y_i^r, \langle \boldsymbol{u}, e_r \otimes \phi(x_i^r)\rangle) + \mu \left(\boldsymbol{u}^\intercal (E \otimes I) \boldsymbol{u}\right), \tag{3.30}$$

Since $E \in \mathbb{R}^{T \times T}$ is positive definite, we can find $B \in \mathbb{R}^{p \times T}, p \geq T$ and $\operatorname{rank} B = T$ such that $E^{-1} = B^\intercal B$, using for example the SVD. Using the properties of the tensor product of linear maps,

$$E^{-1} \otimes I = (B^\intercal B) \otimes I = (B^\intercal \otimes I)(B \otimes I),$$

Consider the change of variable $\boldsymbol{u} = (B^\intercal \otimes I)\boldsymbol{w}$, where $\boldsymbol{w} \in \mathbb{R}^p \otimes \mathcal{H}$. Observe that this can always be done because B is full rank. Rewriting (3.30) using $\boldsymbol{w}$,

$$\begin{aligned} R((B^\intercal \otimes I)\boldsymbol{w}) &= \sum_{r=1}^{T}\sum_{i=1}^{m} \ell(y_i^r, \langle (B^\intercal \otimes I)\boldsymbol{w}, (e_r \otimes \phi(x_i^r))\rangle) + \mu \boldsymbol{w}^\intercal (B^\intercal \otimes I)^\intercal (E \otimes I)(B^\intercal \otimes I)\boldsymbol{w} \\ &= \sum_{r=1}^{T}\sum_{i=1}^{m} \ell(y_i^r, \langle \boldsymbol{w}, (B \otimes I)(e_r \otimes \phi(x_i^r))\rangle) + \mu \boldsymbol{w}^\intercal \boldsymbol{w}, \end{aligned}$$

which is equivalent to

$$S(\boldsymbol{w}) = \sum_{r=1}^{T}\sum_{i=1}^{m} \ell(y_i^r, \langle \boldsymbol{w}, (B_r \otimes \phi(x_i^r))\rangle) + \mu \boldsymbol{w}^\intercal \boldsymbol{w}.$$

We are thus considering a regularized functional $S(\boldsymbol{w})$ where we seek the minimum over functions w in the Hilbert space $\mathbb{R}^p \otimes \mathcal{H}$. Note that in this space the inner product is:

$$\begin{aligned} \langle , \rangle : (\mathbb{R}^p \otimes \mathcal{H}) \times \quad & (\mathbb{R}^p \otimes \mathcal{H}) \to && \mathbb{R} \\ (z_1, \phi(x_1)), \quad & (z_2, \phi(x_2)) \to && \langle z_1, z_2 \rangle \, k(x_1, x_2) \end{aligned}$$

Where $k(\cdot, \cdot)$ is the reproducing kernel of the space of functions $\phi(\cdot)$. However we are only interested in those cases where $z = Be_r = B_r$ for some $r = 1, \ldots, T$, then $\langle B_r, B_s \rangle = E_{rs}^{-1}$. Since the regularizer is clearly increasing in $\|w\|^2$, we can apply the Representer theorem, which states that the minimizer of $S(\boldsymbol{w})$ has the form

$$\boldsymbol{w}^* = \sum_{r=1}^{T}\sum_{i=1}^{m} \alpha_i^r (B_r \otimes \phi(x_i^r)),$$

Using the correspondence between $\boldsymbol{u}^*$ and $\boldsymbol{w}^*$,

$$\boldsymbol{u}^* = B^\intercal \boldsymbol{w}^* = \sum_{r=1}^{T}\sum_{i=1}^{m} \alpha_i^r (\mathrm{vect}\,((\langle B_1, B_r \rangle, \ldots, \langle B_T, B_r \rangle)) \otimes \phi(x_i^r)).$$

Then, we can recover the predictions corresponding to the solutions u_r^* as

$$\begin{aligned} \langle u_r, \phi(\hat{x}^s) \rangle &= \langle \boldsymbol{u}, e_s \otimes \phi(\hat{x}^s) \rangle \\ &= \left\langle \sum_{s=1}^{T}\sum_{i=1}^{m} \alpha_i^r (\mathrm{vect}\,((\langle B_1, B_r \rangle, \ldots, \langle B_T, B_r \rangle)) \otimes \phi(x_i^r)), e_s \otimes \phi(\hat{x}^s) \right\rangle \\ &= \sum_{s=1}^{T}\sum_{i=1}^{m} \alpha_i^r \langle B_s, B_r \rangle \langle \phi(x_i^r), \phi(x_s) \rangle \\ &= \sum_{s=1}^{T}\sum_{i=1}^{m} \alpha_i^r (E^{-1})_{rs} k(x_i^r, \hat{x}^s). \end{aligned}$$

□

Observe that, applying the corresponding feature map $B \otimes I$, the predictions can be obtained equivalently using the common $\boldsymbol{w}$ as

$$\begin{aligned}\langle \boldsymbol{w}, (B \otimes I)(e_s \otimes \phi(\hat{x}^s))\rangle &= \langle \boldsymbol{w}, (B_s \otimes \phi(\hat{x}^s))\rangle \\ &= \sum_{r=1}^{T}\sum_{i=1}^{m} \alpha_i^r \langle B_r \otimes \phi(x_i^r), B_s \otimes \phi(\hat{x}^s)\rangle \\ &= \sum_{r=1}^{T}\sum_{i=1}^{m} \alpha_i^r \langle B_r, B_s\rangle \langle \phi(x_i^r), \phi(\hat{x}^s)\rangle \\ &= \sum_{r=1}^{T}\sum_{i=1}^{m} \alpha_i^r (E^{-1})_{rs} k(x_i^r, \hat{x}^s).\end{aligned}$$

That is, we have expressed the Multi-Task problem as a Single-Task problem with the Multi-Task kernel is

$$\hat{k}(x_i^r, x_j^s) = (E^{-1})_{rs} k(x_i^r, x_j^s).$$

Note that the kernels obtained in this way, unlike those obtained using Lemma 3.27, split the inter-task relations and the similarity between data points. That is, we can implicitly send our data into another, a possibly infinite-dimensional, space and apply the task information after this transformation. This kind of kernels are usually called separable kernels Álvarez et al. (2012) but, to the best of knowledge, this is the first time the they are constructed using tensor products.

Examples of Multi-Task Kernels

Using the framework for Multi-Task learning with Kernel methods we can choose different regularizations, induced by the matrix E, which lead to different Multi-Task approaches.

Independent Tasks The trivial case when $E = I_T$ and therefore $B = I_T$, that is

$$B_r^\intercal = (\overbrace{0}^{1}, \ldots, \overbrace{1}^{r}, \overbrace{0}^{T}),$$

and the kernel is

$$\hat{k}(x_i^r, x_j^s) = \langle B_r, B_s\rangle\, k(x_i^r, x_j^s) = (\delta_{rs}) k(x_i^r, x_j^s).$$

This approach is not a proper MTL method because each task is learned separately, and no coupling is being enforced among tasks.

Independent Parts with Shared Common Model When the matrix B is selected such that its columns are

$$B_r^\intercal = (\overbrace{0}^{1}, \ldots, \overbrace{1}^{r}, \overbrace{0}^{T}, \overbrace{\frac{1}{\mu}}^{T+1}),$$

the corresponding multi-task kernel is:

$$\hat{k}(x_i^r, x_j^s) = \langle B_r, B_s\rangle\, k(x_i^r, x_j^s) = (\frac{1}{\mu} + \delta_{rs}) k(x_i^r, x_j^s)$$

This is equivalent to the approach presented in the work of Evgeniou and Pontil (2004) where it is named *regularized MTL*. The goal is to find a decision function for each task, each being defined by a vector

$$w_r = w + v_r,$$

where w is common to all tasks and v_r is task-specific. The primal problem of *regularized MTL* SVM, using the unified formulation, is

$$\begin{aligned} \underset{w,v_r,\xi_i^r}{\arg\min} \quad & C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\xi_i^r + \frac{1}{2}\langle w, w\rangle + \sum_{r=1}^{T}\frac{\mu}{2}\langle v_r, v_r\rangle \\ \text{s.t.} \quad & y_i^r(\langle w, x_i^r\rangle + \langle v_r, x_i^r\rangle) \geq p_i^r - \xi_i^r, \\ & \xi_i^r \geq 0, \\ \text{for} \quad & r = 1,\ldots,T;\ i = 1,\ldots,m_r. \end{aligned} \tag{3.31}$$

Note that μ is a parameter that controls the tradeoff between the relevance of common and specific models. That is, when μ tends to infinite, the resulting model approaches a common-task standard SVM; when μ tends to zero, a independent task approach is taken, with one standard SVM problem for each task. This is also reflected in the corresponding dual problem

$$\begin{aligned} \underset{\alpha_i}{\arg\min} \quad & \frac{1}{2}\sum_{r,s=1}^{T}\sum_{i,j=1}^{m_r} y_i^r y_j^s \alpha_i^r \alpha_j^s \langle x_i^r, x_j^s\rangle + \frac{1}{2\mu}\sum_{r,s=1}^{T}\sum_{i,j=1}^{m_r} y_i^r y_j^s \alpha_i^r \alpha_j^s \delta_{rs}\langle x_i^r, x_j^s\rangle \\ & - \sum_{r=1}^{T}\sum_{i=1}^{m_r} p_i^r \alpha_i^r \\ \text{s.t.} \quad & 0 \leq \alpha_i^r \leq C \\ \text{for} \quad & r = 1,\ldots,T;\ i = 1,\ldots,m_r. \end{aligned} \tag{3.32}$$

In this dual form, as μ grows, the task-specific part goes to zero, and the most important term is the first one, corresponding to the common part. The opposite effect is obtained when μ shrinks. Moreover, in Evgeniou and Pontil (2004) it is shown that solving (3.31) is equivalent to solving the problem

$$\begin{aligned} \underset{ww_r,\xi_i^r}{\arg\min} \quad & C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\xi_i^r + \frac{1}{2}\sum_{r=1}^{T}\|w_r\|^2 + \frac{\mu}{2}\sum_{r=1}^{T}\left\|w_r - \sum_{s=1}^{T} w_s\right\|^2 \\ \text{s.t.} \quad & y_i^r(\langle w_r, x_i^r\rangle) \geq p_i^r - \xi_i^r, \\ & \xi_i^r \geq 0, \\ \text{for} \quad & r = 1,\ldots,T;\ i = 1,\ldots,m_r. \end{aligned}$$

Now, only the w_r variables are included, and it is clearer that μ penalizes the variance of the w_r vectors, so all models w_r will tend to a common model as μ grows.

This is a very interesting approach because, as it was pointed out in Liang and Cherkassky (2008), this approach has connections with the SVM+ approach from Vapnik and Izmailov (2015).

Graph Laplacian When the tasks are considered as nodes in a graph, and the weights of the edges of this graph portrait the relation between each pair of tasks, the matrix E

can be seen as a Laplacian matrix $L = D - A$. Here A is the adjacency matrix indicating the weights of the edges between each pair of tasks, and D is the degree matrix, a diagonal matrix where each diagonal term is the sum of the corresponding row of A. Observe that using this matrix, the regularization term is

$$\begin{aligned}
\boldsymbol{u}^\intercal (L \otimes I)\boldsymbol{u} &= \sum_{r=1}^{T}\sum_{s=1}^{T} u_r^\intercal L_{rs} u_s \\
&= \sum_{r=1}^{T}\sum_{s=1}^{T} u_r^\intercal (D-A)_{rs} u_s \\
&= \sum_{r=1}^{T}\sum_{s=1}^{T} u_r^\intercal \left(\delta_{rs}\sum_q A_{rq}\right) u_s - \sum_{r=1}^{T}\sum_{s=1}^{T} u_r^\intercal A_{rs} u_s \\
&= \sum_{r=1}^{T} u_r^\intercal \sum_q A_{rq} u_r + \sum_{r=1}^{T} u_s^\intercal \sum_q A_{sq} u_s - \sum_{r=1}^{T}\sum_{s=1}^{T} u_r^\intercal A_{rs} u_s \\
&= \sum_{r=1}^{T}\sum_{s=1}^{T} u_r^\intercal A_{rs} u_s + u_s^\intercal A_{rs} u_s - u_r^\intercal A_{rs} u_s \\
&= \sum_{r=1}^{T}\sum_{s=1}^{T} A_{rs} \left\| u_r - u_s \right\|^2 .
\end{aligned}$$

That is, the distance between task models is penalized, weighted by the degree of similatiy between the tasks as indicated by the graph. Here, the multi-task kernel is defined as

$$\hat{k}(x_i^r, x_j^s) = \langle B_r, B_s \rangle \, k(x_i^r, x_j^s) = \left(L^+\right)_{rs} k(x_i^r, x_j^s).$$

Observe that the pseudoinverse is used because the Laplacian matrices are semipositive definite.

3.4 Multi-Task Learning Methods: An Overview

In this section a general overview of the MTL methods will be shown, categorizing some of the most relevant approaches. We can consider three main groups in this taxonomy: Feature-Based, Parameter-Based and Combination-Based strategies. A section is developed for each group. Most MTL strategies used in deep learning can be categorized as Feature-Based. Also, most kernel methods use Parameter-Based or Combination-Based strategies, but given the relevance of these models, MTL with neural networks and MTL with kernel methods will be treated separately in their own subsections.

3.4.1 Feature-Based MTL

The feature-based methods try to find a set of features that are useful for all tasks. Two main approaches are taken: Feature Learning, which tries to learn new features from the original ones, and Feature Selection which selects a subset of the original features.

Feature Learning approach

Apart from the Multi-Task feedforward Neural Network first shown in Caruana (1997), which will be described later, the first work of Multi-Task Feature Learning is presented in Argyriou et al. (2006). Argyriou et al. assume a multi-task linear model in some RKHS $\mathcal{H}$, where that the task parameters w_t lie in a linear subspace, i.e.

$$\underset{d\times T}{W} = \underset{d\times d}{U}\,\underset{d\times T}{A}$$

where w_t are the columns of W, U is an orthogonal matrix and A is a row-sparse matrix. The minimization problem is

$$\underset{U\in\mathbb{R}^{d\times d},A\in\mathbb{R}^{d\times T}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle U a_r, \phi(x_i^r)\rangle) + \lambda \|A\|_{2,1}^2 \text{ s.t. } U^\intercal U = I. \tag{3.33}$$

where the $L_{2,1}$ regularizer is used to impose row-sparsity across tasks, i.e. forcing some rows of A to be zero while the matrix U is restricted to be orthonormal. Instead of solving (3.33), they show that

$$\underset{U\in\mathbb{R}^{d\times d},A\in\mathbb{R}^{d\times T}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle U a_r, \phi(x_i^r)\rangle) + \lambda \|A\|_{2,1}^2 \text{ s.t. } U^\intercal U = I. \tag{3.34}$$

Although problem (3.33) is not jointly convex in U and A, in Argyriou et al. (2006) and Argyriou et al. (2008) is shown to be equivalent to the convex problem

$$\begin{aligned}\underset{W\in\mathbb{R}^{d\times T},D\in\mathbb{R}^{d\times d}}{\arg\min} &\sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, x_i^r\rangle) + \lambda \sum_{r=1}^{T} \left\langle w_r, D^{-1} w_r\right\rangle \\ \text{s.t. } D \succeq 0,\ &\operatorname{tr}(D) \leq 1.\end{aligned} \tag{3.35}$$

and if (A^*, U^*) is an optimal solution of (3.33), then

$$(W^*, D^*) = \left(U^* A^*, U^* \operatorname{diag}\left(\frac{\|\boldsymbol{a}^1\|_2}{\|A\|_{2,1}}\right), \ldots, \left(\frac{\|\boldsymbol{a}^d\|_2}{\|A\|_{2,1}}\right)(U^*)^\intercal\right)$$

is an optimal solution of problem (3.35); conversely, given an optimal solution (W^*, D^*), if U^* has as columns an orthonormal basis of eigenvectors of D^* and $A^* = (U^*)^\intercal W^*$, (A^*, U^*) is an optimal solution of (3.33). To obtain an optimal solution (W^*, D^*) the authors use a two-step optimization but for a better stability they use the modified problem

$$\begin{aligned}\underset{W\in\mathbb{R}^{d\times T},D\in\mathbb{R}^{d\times d}}{\arg\min} &\sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, \phi(x_i^r)\rangle) + \lambda \sum_{r=1}^{T} \left\langle w_r, D^{-1} w_r\right\rangle + \mu \operatorname{tr}\left(D^{-1}\right) \\ \text{s.t. } D \succeq 0,\ &\operatorname{tr}(D) \leq 1.\end{aligned} \tag{3.36}$$

The optimization with respect to W decouples in each task and the standard Representer Theorem can be used to solve it, and the one with respect to D has a closed solution $D^* = (W^\intercal W + \mu I)^{\frac{1}{2}} / \operatorname{tr}\left((W^\intercal W + \mu I)^{\frac{1}{2}}\right)$. It is interesting to observe that when $\mu = 0$, the regularizer of (3.35) can be expressed as $\operatorname{tr}(W^\intercal D^+ W)$ and by plugging D^* in this

formula we obtain the squared-trace norm regularizer for W:

$$\underset{W\in\mathbb{R}^{d\times T}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, \phi(x_i^r)\rangle) + \lambda \|W\|_*^2 . \tag{3.37}$$

Here, $\|W\|_* = \operatorname{tr}\left((W^\intercal W)^{\frac{1}{2}}\right)$ denotes the trace norm (also known as nuclear norm), which can be seen as the continuous envelope of the rank, thus favouring low-rank solutions of W. That is, the initial problem (3.33) is equivalent to a problem where the trace norm regularization for matrix W is used.

In Argyriou et al. (2007) this idea is extended to any spectral funcion $F : \mathbb{S}_{++}^d \to \mathbb{S}_{++}^d$ where $\mathbb{S}_{++}^d$ is the set of matrices $A \in \mathbb{R}^{d\times d}$ symmetric and positive definite. The definition for the spectral function $F(A)$, where we can diagonalize the matrix A as $V^\intercal \operatorname{diag}(\lambda_1, \ldots, \lambda_d)V$ is

$$F(A) = U^\intercal \operatorname{diag}(f(\lambda_1), \ldots, f(\lambda_d))U .$$

Then, a generalized regularizer for problem (3.35) can be expressed as

$$\sum_t \langle w_t, F(D)w_t\rangle = \operatorname{tr}\left(W^\intercal F(D)W\right) = \operatorname{tr}\left(F(D)WW^\intercal\right) .$$

It is easy to see that problem (3.35) is a particular case where $f(\lambda) = \lambda^{-1}$. In Maurer (2009) some bounds on the excess risks are given for this Multi-Task Feature Learning method.

Another relevant extension is shown in Agarwal et al. (2010), where instead of assuming that the task parameters w_r lie in a linear subspace, the authors generalize this idea by assuming that w_r lies in a manifold $\mathcal{M} \in \mathcal{H}$.

$$\underset{W,\mathcal{M},\boldsymbol{b}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, x_i^r\rangle + b_r) + \lambda \sum_{r=1}^{T} \mathcal{P}_{\mathcal{M}}(w_r), \tag{3.38}$$

where $\mathcal{P}_{\mathcal{M}}(w_t)$ represents the distance between w_t and its projection on the manifold $\mathcal{M}$. Again, an approximation of (3.38) is used to obtain a convex problem and it is solved using a two-step optimization algorithm.

Other distinct, relevant approach for Feature Learning is the one described in Maurer et al. (2013), where a sparse-coding method Maurer and Pontil (2010) is used for MTL. Maurer et al. present the problem

$$\underset{D\in\mathcal{D}_k, A\in\mathbb{R}^{k\times T}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle Da_r, x_i^r\rangle) + \lambda \|D\|_{2,\infty} + \mu \|A\|_{1,\infty} . \tag{3.39}$$

Here, $\mathcal{D}_k$ is the set of k-dimensional dictionaries and every $D \in \mathcal{D}_k$ is a linear map $D : \mathbb{R}^k \to \mathcal{H}$; in the linear case, where $\mathcal{H} = \mathbb{R}^d$, the set $\mathcal{D}_k$ is the set of matrices $\mathbb{R}^{d\times k}$, such that

$$\underset{d\times T}{W} = \underset{d\times k}{D}\,\underset{k\times T}{A} .$$

Although (3.33) and (3.39) share a similar form, there are crucial differences. The matrix U in (3.33) is an orthogonal square matrix, while the matrix D of (3.39) is overcomplete with $k > d$ columns of bounded norm. Also, in problem (3.33) non-linear features can

be used while problem (3.39) is linear. A problem very similar to (3.39) is presented in Kumar and III (2012) where the idea is the same but the regularizers are the $L_{2,2}$ (Frobenius) norm for D and the $L_{1,1}$ norm for A:

$$\underset{D\in\mathcal{D}_k, A\in\mathbb{R}^{k\times T}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle Da_r, x_i^r\rangle) + \lambda \left\|D\right\|_{2,2} + \mu \left\|A\right\|_{1,1}. \tag{3.40}$$

Here, we can interpret this model as a linear sparse combination, encoded in A, of some features encoded in D: $\boldsymbol{w}_r^\intercal \cdot \boldsymbol{x}_i^r = \sum_{i=1}^{k} a_r^i \left(\boldsymbol{D}_i^\intercal \cdot \boldsymbol{x}_i^r\right)$. Unlike the Multi-Task Feature Learning approach of Argyriou et al, this sparse coding formulation is only presented in the linear setting.

Feature Selection approach

The feature selection is also driven by learning a good set of features for all tasks, however it focuses on subsets of the original features. Due to their nature, the works following this strategy are based on linear models. This is a more rigid approach than that of Feature Learning but is also more interpretable.

Most works on Multi-Task Feature Selection uses an $L_{p,q}$ regularization of the weights matrix W. The first work Obozinski et al. (2006) solves the problem

$$\underset{W}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, x_i^r\rangle) + \lambda \left\|W\right\|_{2,1}^2, \tag{3.41}$$

where the $L_{2,1}$ regularization enforces row sparsity and forces different tasks to share a subset of features $\boldsymbol{w}^i$, which are the rows of W. In Liu et al. (2009) the $L_{\infty,1}$ regularization is used for the same goal. Then, in Gong et al. (2012a) this idea is generalized with a capped-$L_{p,1}$ penalty of W, which is defined as $\sum_{i=1}^{d} min(\theta, \left\|\boldsymbol{w}^i\right\|_p)$. That is, the parameter θ enables a more flexible regularization, with small values of θ the smallest rows are pushed towards zero since the rows with norms larger than θ will not dominate the sum, as θ grows this penalty will degenerate to the standard $L_{p,1}$ norm.

In Lozano and Swirszcz (2012) a multi-level lasso selection is presented where the main idea is to decompose each w_r^i, that is, the i-th feature of the r-th task as $w_r^i = \theta^i a_r^i$ and then, using the matrix $\Theta = \operatorname{diag}\left(\theta^1, \ldots, \theta^d\right)$, the y define the problem

$$\underset{\Theta, \boldsymbol{a}_1,\ldots,\boldsymbol{a}_T}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle \Theta \boldsymbol{a}_r, x_i^r\rangle) + \mu \operatorname{tr}\Theta + \nu \left\|A\right\|_{1,1}, \tag{3.42}$$

By doing this, the features i such that $\theta^i = 0$, are discarded for all tasks, but the rest may be shared among tasks or not depending on the values of a_r^i. Observe that this is similar to the sparse coding problem shown in (3.39) with D, the "feature building" matrix, being limited to diagonal matrices since it is acting here as a selection matrix: $\boldsymbol{w}_r^\intercal \cdot \boldsymbol{x}_i^r = \sum_{i=1}^{k} a_r^i \left(\theta_i \cdot \boldsymbol{x}_i^r\right)$.

The feature selection methods based on $L_{p,1}$ regularization are shown to be equivalent to a Bayesian approximation with a generalized Gaussian prior in Zhang et al. (2010). Moreover, this approach also allows to find the relationship among tasks and to identify outliers. In Hernández-Lobato and Hernández-Lobato (2013) a horseshoe prior is used instead to learn feature covariance; and in Hernández-Lobato et al. (2015) this prior is also used to identify outlier tasks.

3.4.2 Parameter-Based MTL

The parameter-based MTL does not focus on shared sets of features across tasks, instead other types of dependencies among the task like the task-parameters w_r, are taken into account. Some approaches rely on the assumption that the Multi-Task weight matrix W has a low rank, others try to learn the pairwise task relations or to cluster the tasks. A different approach is the decomposition one, where the assumption is that the matrix W can be expressed as the summation of multiple matrices. We summarize each approach below.

Low-Rank approach

In the low-rank approach the assumption is that task parameters w_r share a low-dimensional space, or, at least, are close to this subspace. This is similar to the Feature Learning approach, but it is not that rigid, since it allows for some flexibility. The idea in Ando and Zhang (2005) is that the task parameters can be decomposed as

$$w_r = u_r + \Theta^\intercal v_r,$$

where $\Theta \in \mathbb{R}^{k\times d}$ spans a shared low dimensional space, that is $\Theta\Theta^\intercal = I_k$ with $k < d$, and d is the dimension of the data. Under this consideration, the proposed model is

$$\underset{\Theta\in\mathbb{R}^{k\times d},\boldsymbol{u},\boldsymbol{v}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle u_r + \Theta^\intercal v_r, x_i^r\rangle) + \lambda \sum_{r=1}^{T} \|u_r\|^2 \text{ s.t. } \Theta\Theta^\intercal = I_k. \tag{3.43}$$

Observe that this problem shares some similarities with (3.33). However, this is a more flexible approach, since the vectors u_r allow for deviations of the task parameters from the shared subspace. Problem (3.43) is solved using a two-step optimization iterating between minimizing in $\{\Theta, \boldsymbol{v}\}$ and minimizing in $\boldsymbol{u}$. Observe that problem (3.43) is not convex but it can be reformulated as

$$\underset{\Theta\in\mathbb{R}^{k\times d},\boldsymbol{w},\boldsymbol{v}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, x_i^r\rangle) + \lambda \sum_{r=1}^{T} \|w_r - \Theta^\intercal v_r\|^2 \text{ s.t. } \Theta\Theta^\intercal = I_k, \tag{3.44}$$

where the terms $\|w_r - \Theta^\intercal v_r\|^2$ enforces the similarity across tasks by bringing them closer to the shared subspace. In Chen et al. (2009) the following extension is proposed:

$$\underset{\Theta\in\mathbb{R}^{k\times d},\boldsymbol{w},\boldsymbol{v}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, x_i^r\rangle) + \lambda \sum_{r=1}^{T} \|w_r - \Theta^\intercal v_r\|^2 + \mu \sum_{r=1}^{T} \|w_r\|^2 \text{ s.t. } \Theta\Theta^\intercal = I_k. \tag{3.45}$$

And it is shown that (3.45), when relaxing the orthogonality constraint, can be expressed as a convex minimization problem.

A different approach relies on the use of the trace norm of matrix W. This norm penalizes $\sum_{i=1}^{d} \lambda_i(W)^2$, where $\lambda_i(W)$ are the eigenvalues of W, and thus, forcing W to be low-rank. In the work of Pong et al. (2010) new formulations for problems with this trace norm penalty and a primal-dual method for solving the problem is developed. A modification of the trace norm can be found in Han and Zhang (2016), where a capped-trace norm is defined as $\sum_{i=1}^{d} \min(\theta, \lambda_i(W)^2)$. This capped norm, like the capped-L_p norm, can enforce a lower rank matrix for small θ and also degenerates to the trace norm for large enough θ.

Task-Relation Learning approach

In other approaches, like the Feature Learning approach or the Low-Rank approach, the assumption is that all task parameters share the same subspace, which may be detrimental when there exists a negative or neutral transfer. The Task-Relation Learning approach aims to find the pairwise dependencies among tasks and to possibly model positive, neutral and negative transfers between tasks.

One of the first works with the goal of explicitly modelling the pairwise task-relations is Bonilla et al. (2007), where a Multi-Task Gaussian Process (GP) formulation is presented. Assuming a Gaussian noise model $y_i^r \sim N\left(f_r(x_i^r), \sigma_r^2\right)$, Bonilla et al. place a GP prior over the latent functions $\boldsymbol{f}_r$ to induce correlation among tasks:

$$\boldsymbol{f} \sim N\left(\mathbf{0}_{dT}, K^f \otimes K_\theta^{\mathcal{X}}\right)$$

where $\boldsymbol{f} = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_T)$ and $A \otimes B$ is the Kronecker product of two matrices, so $\mathrm{Cov}(f_r(x), f_s(x')) = K_{rs}^f k_\theta^{\mathcal{X}}(x, x')$. That is, instead of assuming a block-diagonal covariance matrix for $\boldsymbol{f}$ as in previous works of Joint Learning Lawrence and Platt (2004), the authors in Bonilla et al. (2007) model the covariance as a product of inter-task covariance and inter-feature covariance. The inference of this model can be done using the standard GP inference for the mean and the variance of the prediction distribution. The mean prediction for a new data point x_* of task s is:

$$f_s(x_*) = (K_{:,l}^f \otimes K^{\mathcal{X}}(:, x_*))^\intercal \Sigma^{-1} \boldsymbol{y},\ \Sigma = K^f \otimes K_\theta^{\mathcal{X}} + \mathrm{diag}(\sigma_1, \ldots, \sigma_T) \otimes I_n.$$

However the interest resides in learning the task-covariance matrix K^f, but this leads to a non-convex problem. The authors propose a low-rank approximation of K^f, which weaken its expressive power. To overcome this disadvantage, in Zhang and Yeung (2010, 2013), using the idea of the Multi-Task GP they consider linear models $f(x_i^r) = \langle w_r, x_i^r \rangle + b_r$ and the prior on matrix $W = (w_1 \ldots, w_T)$ is defined as

$$W|\sigma_r \sim \left(\prod_{r=1}^{T} N\left(\mathbf{0}_d, \theta_i^2 I_d\right)\right) MN\left(0_{d\times m}, I_d \otimes \Omega\right)$$

where $MN\left(M, A \otimes B\right)$ denotes the matrix-variate normal distribution with mean M, row covariance matrix A and column covariance matrix B. It is shown that the problem of selecting the maximum a posteriori estimation of W and the maximum likelihood estimations of Ω and $\boldsymbol{b}$ has a regularized minimization problem that, when relaxing the restrictions on Ω, can be expressed as

$$\begin{aligned} &\underset{\Omega\in\mathbb{R}^{d\times T}, W, \boldsymbol{b}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, x_i^r\rangle + b_r) + \lambda \sum_{r=1}^{T} \|w_r\|^2 + \mu \sum_{r,s=1}^{T} \left(\Omega^{-1}\right)_{rs} \langle w_r, w_s\rangle \\ &\text{s.t. } \Omega \succeq 0, \mathrm{tr}\,\Omega = 1. \end{aligned} \tag{3.46}$$

Other approaches like Argyriou et al. (2013) reach a similar problem from other perspective. Argyriou et al. assume a representation of the structure of the tasks as a graph, then the graph Laplacian in the optimization problem can incorporate the

knowledge about the task structure as shown in Evgeniou et al. (2005):

$$\arg\min_{W,\boldsymbol{b}} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, x_i^r\rangle + b_r) + \mu \sum_{r,s=1}^{T} (L)_{rs} \langle w_r, w_s\rangle , \tag{3.47}$$

where μ is a tuning parameter. Here the regularization can be also written using the adjacency matrix A as

$$\sum_{r,s=1}^{T} \left(L^+\right)_{rs} \langle w_r, w_s\rangle = \sum_{r,s=1}^{T} \left(A^+\right)_{rs} \|w_r - w_s\|^2 ,$$

The goal of Argyriou et al. (2013) and Zhang and Yeung (2010) is to jointly learn the task parameters and the tasks relations. In both cases, they opt for a two-step optimization: one step to learn the task parameters and other to learn the task relations. The first step, according to Evgeniou et al. (2005), the problem (3.47) can be solved in the dual space using a proper kernel

$$k_L(x_i^r, x_j^s) = (L + \lambda I)^{-1}_{rs}\, k(x_i^r, x_j^s).$$

In Zhang and Yeung (2010) the authors propose the use of the trace norm to enforce a low rank task-covariance matrix Θ, which leads to a closed solution. In Argyriou et al. (2013) they want to learn a matrix L that is a valid graph Laplacian, so they propose the following problem in the dual space:

$$\begin{aligned} &\arg\min_{\boldsymbol{\alpha},L} \boldsymbol{\alpha}^\intercal K_L \boldsymbol{\alpha} + \nu \boldsymbol{\alpha}^\intercal \boldsymbol{y} + \operatorname{tr}(L + \lambda I)^{-1} \\ &\text{s.t. } 0 \preceq (L + \lambda I)^{-1} \preceq \frac{1}{\lambda} I,\ (L + \lambda I)^{-1}_{\text{off}} \leq 0,\ (L + \lambda I)^{-1} \mathbf{1}_n = \frac{1}{\lambda} \mathbf{1}_n, \end{aligned} \tag{3.48}$$

where A_{off} denotes the off diagonal entries of A. The restrictions of problem (3.48) are to ensure that L is a valid graph Laplacian, and the trace term in the objective function is to force L to be low-rank so it is easier to find clusters of tasks. The objective function is not jointly convex but is convex in each $\boldsymbol{\alpha}$ and L when we fix the other. For the step to optimize the Laplacian matrix the authors develop an algorithm involving several projection steps using proximal operators. Another work focused on learning the task-relations is Dinuzzo (2013), where the approach the problem a learning problem in an RKHS of vector-valued functions $g : \mathcal{X} \to \mathbb{R}^T$, where the associated reproducing kernel is:

$$H(x_1, x_2) = K_{\mathcal{X}}(x_1, x_2) \cdot L$$

and L is a symmetric positive matrix called *output* kernel. That is, the elements of such RKHS are not real-valued functions but vector-valued functions, where each element of the vector corresponds to a different task.

Task Clustering approach

The Task Clustering approach tries to find K clusters or groups among the original set of T tasks. Usually, the goal is to learn jointly only the tasks in the same cluster, so no negative transfer takes place. The first clustering approach Thrun and O'Sullivan (1996) divides the optimization process in two separate steps: independently learning the task-parameters and jointly learning the clusters of tasks. Using models that involves

distances among points, e.g. kernel methods, they define for each task the distance

$$\operatorname{dist}_{\boldsymbol{\phi}_r}(x, x') = \sqrt{\sum_{i=1}^{d} \phi_r^i \, (x^i - x'^i)^2}.$$

That is, $\boldsymbol{\phi}_r$ parametrizes a distance with a different weight for each feature. Then, for each task $r = 1, \ldots, T$ an optimal weights vector $\boldsymbol{\phi}_r^*$ is computed minimizing the distance between examples of the same class and maximizing the distance among different classes:

$$A_r(\boldsymbol{\phi}_r) = \sum_{s=1}^{T} \delta_{r,s} \sum_{i=1}^{m_r} \sum_{j=1}^{m_s} (y_i^r y_j^s) \operatorname{dist}_{\boldsymbol{\phi}_r}(x_i^r, x_j^r),$$

where δ_{rs} is 1 if r, s are the same and -1 otherwise. After the computation of the optimal parameters $\boldsymbol{\phi}_r^*$ the empirical loss on task r of a model fitted on data of task r using a distance parametrized by $\boldsymbol{\phi}_s^*$ is defined as e_{rs}. Then, the goal is to find clusters B_κ with $\kappa = 1, \ldots, K$ minimizing

$$J(K) = \sum_{\kappa=1}^{K} \sum_{r \in B_\kappa} \frac{1}{|B_r|} \sum_{s \in B_\kappa} e_{r,s},$$

where the number of clusters K with minimum $J(K)$ is selected. That is, the clusters are selected using the results of independently trained tasks and the transfer is done through the parameters $\boldsymbol{\phi}_r$.

From the regularization approach some proposals have also been made. In Jacob et al. (2008) a problem based on Evgeniou and Pontil (2004) is proposed. Considering $U = \frac{1}{T}\mathbf{1}\mathbf{1}^\intercal$, E the $T \times K$ cluster assignment binary matrix, and defining the adjacency matrix $M = E(E^\intercal E)^{-1}E^\intercal$ the problem is

$$\arg\min_{W} \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, x_i^r \rangle) + \lambda(\mu_{\mathrm{m}} \Omega_{\mathrm{m}}(W) + \mu_{\mathrm{b}} \Omega_{\mathrm{b}}(W) + \mu_{\mathrm{w}} \Omega_{\mathrm{w}}(W)), \tag{3.49}$$

where $\Omega_{\mathrm{m}} = \operatorname{tr}(WUW^\intercal)$ is the mean regularization, $\Omega_{\mathrm{b}} = \operatorname{tr}(W(M-U)W^\intercal)$ is the inter-cluster variance regularization and $\Omega_{\mathrm{w}} = \operatorname{tr}(W(I-M)W^\intercal)$ is the intra-cluster variance regularization. This problem cannot be solved using the results of Evgeniou et al. (2005) because the regularization used is not convex, so a convex relaxation is needed.

A similar approach is presented in Kang et al. (2011) where, using the results from Argyriou et al. (2008), they propose a trace norm regularizer of the matrices $W_\kappa = WQ_\kappa$, where Q_κ are $T \times T$ binary, diagonal matrices where r-th element of the diagonal indicates if task r corresponds to cluster κ. They consider the problem

$$\begin{aligned} \arg\min_{W, Q_1, \ldots, Q_T, \boldsymbol{b}} \quad & \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r, x_i^r \rangle + b_r) + \lambda \sum_{\kappa=1}^{K} \|WQ_\kappa\|_*^2 \\ \text{s.t.} \quad & \sum_{\kappa=1}^{K} Q_\kappa = I \text{ with } 0 \leq Q_{\kappa t} \leq 1 \end{aligned} \tag{3.50}$$

Here, the trace norm acts on each cluster, so it enforces that the matrices of vectors w_r in

the same cluster have low-rank. This can be seen as a clusterized version of Multi-Task Feature Learning Argyriou et al. (2006, 2008), that is, instead of assuming that all tasks share the same subspace, only the tasks in the same cluster do, however the explicit learned features cannot be recovered. The optimization of matrices Q_κ on problem (3.50) is done by reparametrizing these matrices as

$$Q_\kappa[r,r] = \frac{\exp(\alpha_{\kappa r})}{\sum_{g=1}^{K}\exp(\alpha_{gr})}$$

and performing gradient descent over the $\alpha_{\kappa r}$ of the regularizer $\|WQ_\kappa\|_*^2$.

Another approximation to clusterized MTL is provided in Crammer and Mansour (2012), where a two-step procedure if described as follows. Considering that K initial clusters are fixed containing the T tasks, then two steps are repeated: First K single task models f_κ are fitted using the pooled data from tasks in cluster κ. Secondly each task r is assigned to the cluster κ whose function f_κ obtains the lowest error in task r. The proposal of Barzilai and Crammer (2015) takes the idea of the cluster assignation step from Crammer and Mansour (2012) and is also inspired by the sparse coding work of Kumar and III (2012). In this work the weights matrix is $W = DA$, where $D \in \mathbb{R}^{d\times K}$ contains as columns the hypotheses for each cluster and $G \in \mathbb{R}^{K\times T}$ is the task assignment matrix, that is $g_r = G_r \in \{0,1\}$ and $\|g_r\|^2 = 1$. The corresponding optimization problem is

$$\begin{aligned}&\underset{D\in\mathbb{R}^{d\times K}, G\in\mathbb{R}^{K\times T}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r}\ell(y_i^r, \langle Dg_r, x_i^r\rangle) + \lambda\|D\|_{2,1}\\ &\text{s.t. } g_r \in [0,1],\ \|g_r\|^2 = 1,\end{aligned} \tag{3.51}$$

where the constraints on g_r have been relaxed to be in the $[0,1]$ interval in order to make problem (3.51) convex. Observe that (3.51) is similar to (3.40) but different restrictions are used to ensure that G can be seen as a clustering assignment matrix.

Decomposition approach

The Decomposition approach considers that assuming that the task parameters resides in the same subspace or that the parameter matrix W is too restrictive for real world scenarios. The proposition is then to decompose the parameter matrix in the sum of two matrices, i.e. $W = U + V$ where usually U captures the shared properties of the tasks and V accounts for the information that cannot be shared among tasks. This models also receive the name of *dirty models* because they assume that the data is *dirty* and cannot be constrained to rigid subspaces. The optimization problem is

$$\underset{U,V\in\mathbb{R}^{d\times T}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r}\ell(y_i^r, \langle u_r + v_r, x_i^r\rangle) + \lambda g(U) + \mu h(V), \tag{3.52}$$

where $g(U)$ and $h(V)$ are different regularizers for U and V, respectively. In Jalali et al. (2010) the authors use $g(U) = \|U\|_{1,\infty}$ to enforce block-sparsity and $h(V) = \|V\|_{1,1}$ to enforce element-sparsity. In Chen et al. (2010) $g(U) = \|U\|_{2,2}$ to enforce low-rank while maintaining $h(V) = \|V\|_{1,1}$. In Chen et al. (2011) both regularizers seek properties shared among all tasks, $g(U) = \|U\|_{2,2}$ to enforce a low-rank and $h(V) = \|V\|_{1,2}$ for row-sparsity. In Gong et al. (2012b) they propose $g(U) = \|U\|_{1,2}$ to enforce row-sparsity, i.e. the tasks

share a common subspace; and $h(V) = \|V\|_{1,2}$ which penalizes the orthogonal parts to the common subspace of task-parameter, the authors state that it penalizes outlier tasks.

Other approaches generalize the decomposition method by assuming that the parameter matrix can be expressed as $W = \sum_{l=1}^{L} W_l$, then the problem to solve has the form

$$\underset{W_1,\dots,W_L \in \mathbb{R}^{d\times T}}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \left\langle \sum_{l=1}^{L} (W_l)_r, x_i^r \right\rangle) + \sum_{l=1}^{L} \lambda_l r(W_l), \tag{3.53}$$

In Zweig and Weinshall (2013) the regularizer used is $r(W_l) = \|W_l\|_{2,1} + \|W_l\|_{1,1}$ to enforce the row and element-sparsity. In Han and Zhang (2015) the regularizer is $r(W_l) = \sum_{r,s=1}^{T} \|(W_l)_r - (W_l)_s\|^2$, which alongside some constraints it allows to build a tree of task groups, where the root contains all the tasks and the leafs only correspond to one task.

3.4.3 Combination-based

The combination-based methods use a combination of task-specific models and models that are common to all tasks. These two models are learned simultaneously with the goal of leveraging the common and specific information.

The first proposal of the frequentist approach, which uses the SVM as the model, is found in Evgeniou and Pontil (2004) where the *regularized MTL* SVM is presented. The goal is to find a decision function for each task, each being defined by a vector $w_r = w + v_r$ and a bias b_r. Here w is common to all tasks and v_r is task-specific. Instead of imposing some restrictions such as low-rank or inter-task regularization the idea is to impose the coupling by directly placing a model w that is common to all tasks. The v_r part is added so each model can be adapted to the specific task.

Multiple extensions of the work of Evgeniou and Pontil (2004) have been presented: in Li et al. (2015); Xu et al. (2014) the method is extended to the Proximal SVM Fung and Mangasarian (2001) and Least Squares SVM Suykens and Vandewalle (1999), respectively. Also, in Parameswaran and Weinberger (2010) the idea is adapted for the Large Margin Nearest Neighbor model Weinberger and Saul (2009). However, in this work we are interested mainly in two extensions: one is the work of Evgeniou et al. (2005) and the other is developed in Cai and Cherkassky (2009); Liang and Cherkassky (2008), both will be described in detail in Section 3.3. We can express the optimization problem as

$$\underset{w,V}{\arg\min} \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(y_i^r, \langle w_r + v_r, x_i^r\rangle) + \lambda(\mu \|w\|^2 + \operatorname{tr}(V^\intercal V)), \tag{3.54}$$

where μ is a tradeoff parameter to leverage the common and task-specific information and V is the matrix with the specific vectors v_r as columns. When the dual problem of (3.54) is obtained, the result is a dual problem where the kernel encodes this combination of common and shared parts, that is the kernel function has the form

$$k(x_i^r, x_j^s) = \mu_1 k(x_i^r, x_j^s) + \mu_2 \delta_{rs} k(x_i^r, x_j^s)$$

where μ_1, μ_2 are scalar parameters and δ_{rs} is 1 only if r and s are the same task, so it encodes the specific part of the model. In the linear case, this is equivalent to a feature extension strategy, like the domain adaptation scheme proposed in III (2007) with only target and source domain, but that can be easily be adapted to MTL.

3.4.4 Multi-Task Learning with Neural Networks

Deep learning has suffered an enormous development over the last decade, with multiple variants having great success in many fields and applications such as Convolutional Neural Networks for image recognition, Generative Adversarial Networks for generative models or Transformers for Natural Language Processing. The feature-learning process natural to neural networks is usually the main idea behind multi-task learning strategies but not always. In this subsection we explore some of these strategies and their connection with other methods used in kernel or linear models.

As proposed in Ruder (2017) we can divide the Multi-Task approaches with neural networks in hard sharing and soft sharing. The hard sharing approach is the most common one, and it consists on sharing between all tasks the hidden layers of the network while keeping some task-specific layers. The first example dates back to Caruana (1997) where only the output layers are task-specific. The goal of this approach is to learn a set of features that is useful for all tasks simultaneously, so the transfer of information between tasks is done in this feature building process. Although this approach has some theoretical properties to improve learning Baxter (2000), it is a very rigid approach since it assumes that the learned features and the feature learning process must be the same for all tasks.

The soft sharing approach to Multi-Task learning tries to tackle these problems with different proposed strategies. In Cavallanti et al. (2010) the authors propose to learn one perceptron for each task and the update of each perceptron is determined by an interaction matrix A such that

$$w_r(\tau+1) = w_r(\tau) - \nu y_t A^{-1}_{r,s} x^s_i \tag{3.55}$$

where w_r are the parameters of the perceptron corresponding to task r. That is, the weights of task r are updated when a mistake is committed on task s and the update is scaled according to the position r, s of matrix A^{-1}. This approach only uses perceptrons, so its expressive power is quite limited. Also, the matrix A has to be defined a priori and is not learned from data. In Long and Wang (2015) they propose a model to overcome these limitations. They use a multi-task architecture for computer vision is presented with multiple shared layers and multiple task-specific layers. Moreover, the use a Bayesian approach to learn the task relationships. They place a tensor prior over the network parameters such that in each specific layer l, there are T matrices W^l_r for $r = 1, \dots, T$ and if we denote as $\mathcal{W}^l$ as the vector (tensor) of such matrices, then

$$p(\mathcal{W}^l) = TN\left(0, \Sigma_1, \Sigma_2, \Sigma_3\right),$$

where Σ_1, Σ_2 model the row-covariance and task-covariance in each matrix W^l_r and Σ_3 model the intertask covariance. The parameters $\mathcal{W}^l$ are learned automatically by using gradient descent and for the covariance matrices a Maximum Likelihood Estimator is used afer each epoch. If we take a look at the update rule for the network parameters

$$W^l_r(\tau+1) = W^l_r(\tau) - \nu\left(\frac{\partial \ell(f_r(x^r_i), y^r_i)}{\partial W^l_r} + \left[(\Sigma_1 \otimes \Sigma_2 \otimes \Sigma_3)^{-1} \text{vect}\left(\mathcal{W}^l\right)_{:,:,r}\right]\right)$$

where ℓ is the loss function, we can observe some similarities with the update (3.55), where the inverse of the Kronecker product of covariances model how each task affect the others. Although this approach uses learns the matrix relations, the architecture is still restrictive since it assumes that all tasks can share a number of hidden layers.

In Misra et al. (2016) they propose a strategy named "Cross-stitch" networks which uses one network for each task, but these networks are connected using a linear combination of the outputs of each layer. Considering a case of multi-task learning with two tasks 1 and 2, if h_1^l, h_2^l are the output values of the l-th layers of the networks of tasks 1 and 2, respectively, then these outputs are combined as

$$\begin{bmatrix} \tilde{h}_1^l \\ \tilde{h}_2^l \end{bmatrix} = A^l \begin{bmatrix} h_1^l \\ h_2^l \end{bmatrix}, = \begin{bmatrix} \alpha_{11}^l & \alpha_{12}^l \\ \alpha_{21}^l & \alpha_{22}^l \end{bmatrix} \begin{bmatrix} h_1^l \\ h_2^l \end{bmatrix}, \tag{3.56}$$

where the 2×2 matrix A^l is a "cross-stitch" unit and defines the linear combination to compute $\tilde{h}_1^l, \tilde{h}_2^l$ which will be the input values for the $l+1$-th layer. The network parameters and the "cross-stitch" values α can both be learned using backpropagation. If $A^l = I_{2\times 2}$ in all layers this is equivalent to two independent networks, one for each task. While using constant matrices as "cross-stitch" units results in two identical common networks. This strategy is extended in Ruder et al. (2017), where the authors include two modifications: network parameters of each network are divided in two spaces, each expecting to capture different properties of the data, and there are learnable skip-connections for each task. The first modification implies that the number of parameters is doubled, that is in each task-specific network and in each layer l there are 2 outputs for each task: $h_{r,a}^l, h_{r,b}^l$, each obtained with a different matrix of parameters $W_{r,a}^l, W_{r,b}^l$. Then, in the combination (3.56), A is a 4×4 matrix, because each $\alpha_{:,:}^l$ is a 2×2 matrix. The matrix A not only determines how the tasks are related but how each space of each task is related with the rest. To enforce that each space capture different properties they use the regularizers

$$\left\| (W_{r,a}^l)^\intercal W_{r,b}^l \right\|_{2,2}^2,$$

where the $L_{2,2}$ (Frobenius) norm is typically the differentiable substitute used for the rank, so the parameters matrices $W_{r,a}^l$ and $W_{r,b}^l$ span orthogonal spaces. Particular values of the matrices A^l can result in a Combination-based approach where there is a shared common model and task-specific ones. The skip-connections are reflected in a final layer in each network that receives as input the linear combination of the activation values $h_{r,1}^l$ and $h_{r,2}^l$ for every layer l. This linear combination uses learnable parameters β_r for each task.

Other approaches consider tensor-based methods instead of the "cross-stitch" networks to learn the networks parameters and the degree of sharing between tasks from data. In Yang and Hospedales (2017a) the authors propose a tensor-generative strategy to model the the parameters of each layer. If W_r^l is the $d_1^l \times d_2^l$ parameter matrix for task r in layer l, in each layer we can consider the collection of such matrices as a $d_1^l \times d_2^l \times T$ tensor $\mathcal{W}^l$. We can build these tensors from other pieces in different ways, for example consider a latent basis matrices $d_1^l \times d_2^l \times K$ tensor $\mathcal{L}$ and the $K \times T$ matrix S, then

$$W_r^l = \mathcal{W}_{:,:,r} = \sum_{\kappa=1}^{K} \mathcal{L}_{:,:,\kappa} S_{\kappa,r}.$$

That is, all task parameters are linear combinations of the latent-basis matrices defined in $\mathcal{L}$. Observe that this is a generalization of the sparse coding scheme presented in III (2009) and shown in equation (3.39). Since all the strategies to build $\mathcal{W}$ from other pieces is based on tensor products, the whole process is differentiable and all those pieces can be learned using gradient descent. The other tensor-based approach is presented in Yang

and Hospedales (2017b) where, instead of a tensor-building approach, they consider the tensors $\mathcal{W}^l$ as the collection of matrices W_r^l and use tensor-trace norms to enforce the coupling between different tasks. Since the tensor-trace norms is not differentiable, they use the subgradient for the backpropagation during training. This approach can be seen as a low-rank can be categorized as low-rank, that is, Parameter-Based.

All the previous approaches consider the same architecture for each task, although the skip-connections can alleviate this, the sharing of information is still made in a layer-wise manner, so the same architecture has to be considered for every network. In Sun et al. (2020) they propose a single network with L layers and with skip-connections between all layers, so each task can use a specific policy. That is, task 1 can use the first, third and fourth layer while task 2 might use the second and fourth alone. In this example, the first and third are specific for task 1, the second layer is specific for task 2 while the third and fourth are shared layers. In general, there is a binary $L \times T$ policies matrix U that determines which layers are used for each task. The problem is a bi-level optimization

$$\min_{U} \min_{W_1,\dots,W_L} \ell(U, W_1, \dots, W_L).$$

The optimization with respect the network parameters is done using back-propagation, while the optimization with respect to U uses a specific algorithm.

3.4.5 Multi-Task Learning with Kernel Methods

Kernel Methods are also models that have been successful in a lot of areas. The principal appeal of these models is that the original features are implicitly transformed to a kernel space, possibly infinite-dimensional, where the problems of classification or regression for example are usually easier to solve.

Unlike deep learning models, the kernel features used are not learnable, but are implicitly defined a priori by the kernel function used like Gaussian, polynomial or Matérn kernels for example. This fact makes it more difficult to use a feature-based MTL approach with kernel models. Instead, other of the reviewed approaches can be taken.

One important approximation to MTL with kernel models is learning vector-valued functions, in which the target space is not scalar but a vector one. That is, the sample data X, Y are pairs $(x_i, \boldsymbol{y}_i)$ where $x_i \in \mathcal{X}$, e.g. $\mathcal{X} = \mathbb{R}^d$ and $y_i \in \mathcal{Y}^T$. This approach finds its motivation in multi-output learning, where multiple targets are learned are defined for each input. Using kernel models the multi-output regularized risk is

$$\sum_{i=1}^{n} \ell(W^\intercal \phi(\boldsymbol{x}_i) + \boldsymbol{b}, \boldsymbol{y}_i) + \mu\Omega(W), \tag{3.57}$$

where W is the matrix with T columns, one for each output and $\boldsymbol{b}$ is the vector of corresponding biases. A commonly used loss function ℓ is

$$\ell(W^\intercal \phi(\boldsymbol{x}_i) + \boldsymbol{b}, \boldsymbol{y}_i) = \|W^\intercal \phi(\boldsymbol{x}_i) + \boldsymbol{b} - \boldsymbol{y}_i\|_2^2$$

and Ω is a regularizer for matrix W that can enforce different behaviours, for example the trace norm regularizer is used for enforcing a low rank matrix. In Micchelli and Pontil (2004, 2005) the authors develop the theory for operator-valued kernels, that are the kernel functions corresponding to vector-valued functions. They show an extension of the representer theorem for operator-valued kernels when a Tikhonov regularization for vector-valued functions is used. Although these are interesting results, they do not

provide explicit algorithms to learn such functions. Moreover, multi-task learning is not that well suited for this formulation, since for each data x_i^r there is a single scalar y_i^r. The multi-task regularized risk using this formulation is

$$\sum_{i=1}^{n} \ell(A \odot W^\intercal \phi(\boldsymbol{x}_i) + \boldsymbol{b}, \boldsymbol{y}_i) + \mu\Omega(W), \tag{3.58}$$

where A is a binary matrix with the same sparsity pattern that Y.

Other approach is a feature-based strategy presented in Argyriou et al. (2008) presented in subsection 3.4.1 whose optimization problem is (3.33), and the corresponding clusterized extension of Kang et al. (2011) whose corresponding problem is shown in (3.50).

The task relation learning is also a common approach to MTL with kernel models. Using a Gaussian Process formulation, different strategies to learn the task-covariance matrix are shown in Bonilla et al. (2007); Lawrence and Platt (2004). Also, regularized problems are proposed using a inter-task regularization that penalize

$$\sum_{r,s=1}^{T} \left(L^+\right)_{rs} \langle w_r, w_s \rangle = \sum_{r,s=1}^{T} \left(A^+\right)_{rs} \|w_r - w_s\|^2,$$

is used in Zhang and Yeung (2010) and Argyriou et al. (2013), as shown in problems (3.46) and (3.47), respectively.

Multi-Task Problems

The concept of Multi-Task problem is not clear because different definitions have been considered in the literature. In this work only supervised problems will be considered, so we will refer to them just as problems.

Multiple kind of problems can be faced with a Multi-Task Learning approach. The most common definition of Multi-Task problem is a homogenenous setting, where all tasks are sampled from the same space. That is, we have a space $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ is the feature space and $\mathcal{Y}$ is the target space, which can be $\mathbb{R}$ in the regression case or $\{0, 1\}$ in the classification one, and each task has a possibly different distribution $P_r(x, y)$ over this shared space. In this type of problems, all the tasks have the same number of features and the same target space, that is, every task is either a regression or classification problem, there cannot be a mix of regression and classification tasks. There are other heterogeneous definitions where each task can be sampled from a different space $\mathcal{X}_r \times \mathcal{Y}_r$, and, therefore, a mix of regression or classification tasks can be considered. However, in this work we are only interested in the homogenenous case.

Within the homogenenous MTL problems we can consider the following cases, depending on the nature of each task and the task definition procedure:

- Pure Multi-Task Problems. These are problems where each task has been sampled from a possibly different distribution, so we have possibly different samples (X_r, Y_r).
  - MT single-reg: This is the case where each task is a regression problem with a single target, e.g. $\mathcal{Y} = \mathbb{R}$.
  - MT bin-clas: This is the case where each task is a binary classification problem , e.g. $\mathcal{Y} = \{0, 1\}$.

  - MT multi-reg: This is the case where each task is a regression problem with multiple targets, e.g. $\mathcal{Y} = \mathbb{R}^m$, where m is the number of targets.
  - MT multi-clas: This is the case where each task is a multi-class classification problem, e.g. $\mathcal{Y} = \{0, 1, \ldots, C\}$, where C is the number of classes.

- Artificial Multi-Task Problems: These are problems that can be seen as Multi-Task and be solved using MTL strategies, but it is not the only way to solve them. The main difference is that although the target samples might be different across tasks, the set of features sampled is shared by all tasks, i.e. $(X_r, Y_r) = (X, Y_r)$.
  - multi-reg: This is the case where a m-target regression problem is converted into a multi-task problem by replicating m times the features X and using one of the targets for each repetition, so we have m single target regression problems, each considered a different task.
  - multi-clas: This is the case where a multi-class classification problem with C classes is converted into a multi-task problem by replicating C times the features X and considering a one vs all scheme for each repetition with a different positive class in each one, so we have C binary classification problems, each considered a different task.

In Table 3.1 we show some of the most used multi-task learning problems, exposing its characteristics and the papers that use them.

Name	n	d	T	Nature	References
school	15362	27	139	MT single-reg	Evgeniou and Pontil (2004) Evgeniou et al. (2005) Argyriou et al. (2006, 2008, 2007) Bonilla et al. (2007) Zhang and Yeung (2010) Agarwal et al. (2010) Chen et al. (2011) Zhou et al. (2011) Gong et al. (2012b) Kumar and III (2012) Zhang and Yeung (2013) Han and Zhang (2016) Jeong and Jun (2018)
20-newsgroup	18000	t.v.	20	multi-clas	Ando and Zhang (2005) III (2009)
Reuters-RCV1	800000	t.v.	103	multi-clas	Yu et al. (2005) Ando and Zhang (2005)
computer	3600	13	180	MT single-reg	Argyriou et al. (2006, 2008, 2007) Evgeniou et al. (2005) Agarwal et al. (2010) Kumar and III (2012) Jeong and Jun (2018)
landmine	14820	10	29	MT bin-clas	Xue et al. (2007) Jebara (2011) III (2009) Jawanpuria and Nath (2012)
MHC-I	32302	184	47	MT bin-clas	Jacob et al. (2008) Jawanpuria and Nath (2012)
dermatology	366	33	6	multi-clas	Jebara (2004) Argyriou et al. (2008)
sentiment	2000	473856	4	MT bin-clas	III (2009) Zhang and Yeung (2010) Crammer and Mansour (2012) Zhang and Yeung (2013) Barzilai and Crammer (2015)
sarcos	44484	21	7	multi-reg	Zhang and Yeung (2010)

Continued on next page

Table 3.1 – continued from previous page

Name	n	d	T	Nature	References
					Chen et al. (2011) Zhou et al. (2011) Jawanpuria and Nath (2012) Zhang and Yeung (2013) Ciliberto et al. (2015)
isolet	7797	617	5	MT multi-clas	Parameswaran and Weinberger (2010) Gong et al. (2012a)
mnist	70000	400	10	multi-clas	Kang et al. (2011) Kumar and III (2012) Zweig and Weinshall (2013) Jeong and Jun (2018)
usps	9298	256	10	multi-clas	Kang et al. (2011) Kumar and III (2012) Zweig and Weinshall (2013) Jeong and Jun (2018)
adni	675	306	6	MT single-reg	Gong et al. (2012a) Gong et al. (2012b)
microarray	131	21	19	multi-reg	Lozano and Swirszcz (2012) Han and Zhang (2016)
cifar10	50000	1024	10	multi-clas	Zweig and Weinshall (2013) Han and Zhang (2016)
parkinson	5875	19	42	MT single-reg	Jawanpuria and Nath (2012) Jeong and Jun (2018)

TABLE 3.1: Multi-Task Learning Problems. The columns show the number of samples n, the dimension d, the number of tasks T and the nature of the problem.

3.5 Conclusions

In this chapter, we covered...

Chapter 4

A Convex Formulation for Multi-Task Learning

4.1 Introduction

4.2 Convex Multi-Task Learning with Kernel Models

As explained in the previous chapters, kernel models offer many good properties such as implicit transformation to a possibly infinite-dimensional space and convexity. In thise models a regularized risk problem is solved. A general formulation for kernel models is

$$\hat{R}_z(\boldsymbol{w}) := \sum_{i=1}^{n} \ell(\boldsymbol{w}^\intercal \phi(\boldsymbol{x}_i) + b, y_i) + \mu\Omega(\boldsymbol{w}), \tag{4.1}$$

where z is the sample $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ and $\Omega(w)$ is a regularizer for $\boldsymbol{w}$, tipically the L_2 norm: $\|w\|^2$. Observe that b, the bias term, is not regularized since it does not affect the capacity of the hypothesis space. In (4.1) ϕ is a fixed transformation function such that there exists a "kernel trick", that is a kernel function k for which

$$\langle \phi(x), \phi(y) \rangle = k(x, y).$$

These models embrace the Structural Risk Minimization paradigm by limiting the capacity of the space of hypothesis. This is done by using the L_2 norm of w as regularizer, which is equivalent to limiting our space of candidates to vectors inside a ball of some radius.

Multi-Task Learning with kernel models require imposing some kind of coupling between models in the learning process. The feature learning or feature sharing approach, which is usually adopted with neural networks, is not feasible when using kernel models, since the (implicit) transformation functions used are not learned but fixed. Therefore, other strategies have to be developed. One of the first approaches to MTL with kernel models was developed in Evgeniou and Pontil (2004), where the models for each task are defined as:

$$\boldsymbol{w}_r = \boldsymbol{w} + \boldsymbol{v}_r,$$

where w is a common part, shared by all models, and v_r is a task-specific part. With this approximation, the transfer of information is performed by the common part $\boldsymbol{w}$. The

regularized risk that is minimized is

$$R_{\boldsymbol{z}}(\boldsymbol{w}) := \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(\boldsymbol{w}^\intercal \phi(\boldsymbol{x}_i) + \boldsymbol{v}_r^\intercal \phi_r(\boldsymbol{x}_i), y_i) + \mu_c \left\|\boldsymbol{w}\right\|^2 + \mu_s \sum_{r=1}^{T} \left\|\boldsymbol{v}_r\right\|^2, \tag{4.2}$$

where μ_c and μ_s are the hyperparameters to control the common and specific parts regularization, respectively. Here $\boldsymbol{z}$ is a MTL sample

$$\boldsymbol{z} = \bigcup_{r=1}^{T} \{(x_1^r, y_1^r), \ldots, (x_m^r, y_m^r)\}.$$

Observe also that in (4.2) different transformations are used. The transformation ϕ corresponds to common part of the model, while ϕ_r is task-specific. This is a joint learning approach that is developed for the L1-SVM, which Evgeniou et al. give the name of *Regularized MTL* but we will refer to as *additive* approach. Observe that as $\frac{\mu_c}{\mu_s} \to \infty$ we would have a common part $\boldsymbol{w}$ that tends to zero, which would results in independent models for each task, i.e. $\boldsymbol{w}_r \approx \boldsymbol{v}_r$. On the contrary, when $\frac{\mu_c}{\mu_s} \to 0$, the task-specific parts tend to zero and every model is the common part, i.e. $\boldsymbol{w}_r \approx \boldsymbol{w}$. There are two asymptotical behaviours: the first one tends to an ITL approach, while the second one tends to a CTL one. The MTL formulation is one strategy that lies between those two approaches, CTL and ITL, combining them to achieve a more flexible model.

The asymptotical properties of this approach offer an interpretation to understand the influence of each hyperparameter, but they are not applicable in practice. In Ruiz et al. (2019) an alternative formulation for this joint learning approach is proposed. The models for each task are defined as a convex combination of the common and task specific parts:

$$\boldsymbol{w}_r = \lambda \boldsymbol{w} + (1 - \lambda)\boldsymbol{v}_r,$$

where λ is a hyperparameter such that $\lambda \in [0, 1]$. The corresponding regularized risk is

$$R_{\boldsymbol{z}}(\boldsymbol{w}) := \sum_{r=1}^{T}\sum_{i=1}^{m_r} \ell(\boldsymbol{w}^\intercal \phi(\boldsymbol{x}_i) + \boldsymbol{v}_r^\intercal \phi_r(\boldsymbol{x}_i), y_i) + \mu_c \left\|\boldsymbol{w}\right\|^2 + \mu_s \sum_{r=1}^{T} \left\|\boldsymbol{v}_r\right\|^2,$$

We will name this approach *convex*, in contrast to the *additive* approach of the original formulation. With this formulation, the interpretation of λ is straight-forward. The model with $\lambda = 1$ is equivalent to learning a single common model for all tasks, that is $\boldsymbol{w}_r = \boldsymbol{w}$. When $\lambda = 0$, the models for each task are completely independent: $\boldsymbol{w}_r = \boldsymbol{v}_r$. The convex formulation also define an MTL model that is between a CTL approach and an ITL one, but it presents two advantages: the values of λ that recover the CTL and ITL approaches are known, and these values are specific, not an asymptotical behaviour as in the original formulation. Moreover, it is shown in the paper that the two formulations, the *additive* and *convex*, are equivalent with an L1-SVM setting. The proposal of Ruiz et al. (2019) is made only for L1-SVM, but it is extended to L2 and LS-SVMs in Ruiz et al. (2021). In this subsection we present the convex formulation for L1, L2 and LS-SVM, as well as the equivalence results between the *additive* and *convex* formulations.

4.2.1 L1 Support Vector Machine

The L1-SVM Vapnik (2000) is the original and most popular variant of the SVMs and is also the basis of the MTL formulation in Evgeniou and Pontil (2004). I will present

the development for the *additive* approach and the one for the *convex* using an L1-SVM setting. Then I will show the equivalence between the two approaches and discuss its differences.

4.2.1.1 *additive* MTL L1-SVM

The *additive* MTL primal problem formulation, presented in Evgeniou and Pontil (2004) and extended for task-specific biases in Cai and Cherkassky (2012), is

$$\begin{aligned} \underset{w,\boldsymbol{v}_r,\xi}{\arg\min} \quad & J(\boldsymbol{w},\boldsymbol{v}_r,\xi) = C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\xi_i^r + \frac{1}{2}\sum_{r=1}^{T}\|\boldsymbol{v}_r\|^2 + \frac{\mu}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \quad & y_i^r(\boldsymbol{w}\cdot\phi(x_i^r) + b + \boldsymbol{v}_r\cdot\phi_r(x_i^r) + d_r) \geq p_i^r - \xi_i^r, \\ & \xi_i^r \geq 0;\ i = 1,\dots,m_r,\ r = 1,\dots,T. \end{aligned} \tag{4.3}$$

The prediction model is then

$$h_r(\boldsymbol{x}) = \boldsymbol{w}\cdot\phi(x_i^r) + b + \boldsymbol{v}_r\cdot\phi_r(x_i^r) + d_r$$

for regression and

$$h_r(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{w}\cdot\phi(x_i^r) + b + \boldsymbol{v}_r\cdot\phi_r(x_i^r) + d_r\right)$$

for classification. Observe again that the transformation ϕ is used for the common part and is shared by all tasks, while the transformation ϕ_r is task-specific.

In (4.3) there are two kind of hyperparameters: C_r and μ, which, in combination, balance the different parts of the objective function. Hyperparameter C plays the same role than in the standard L1-SVM : it balances the tradeoff between the loss incurred by the model, represented by the hinge variables ξ_i^r and complexity of the models, represented by the norms $\|\boldsymbol{w}\|$ and $\|\boldsymbol{v}_r\|$. Large values of C highly penalize the loss, so the resulting models are more complex because they have to adapt to the training sample distribution, but these models generalize worse. Small values of C penalize more the norms of w and v_r so the resulting models are simpler but not so dependent on the training sample.

Hyperparameter μ, in combination with C, balances the specifity of our models. Large values of μ, penalize the common part, resulting in more specific models; while small values of μ, alongside large values of C, result in a vanishing regularization of the specific parts which leads to common models. We can find the following cases:

- Reduction to an ITL approach:

$$\mu \to \infty \implies h_r(\boldsymbol{x}_i^r) = \boldsymbol{v}_r\cdot\phi_r(x_i^r) + d_r.$$

  That is, the models are learned independently because the common part vanishes.

- Reduction to a CTL approach:

$$C \to 0, \mu \to 0 \implies h_r(\boldsymbol{x}_i^r) = \boldsymbol{w}\cdot\phi(x_i^r) + b.$$

  That is, the model for all tasks is common because the specific parts disappear.

- Pure MTL approach:

$$\mu_{\text{inf}} < \mu < \mu_{\text{sup}} \implies h_r(\boldsymbol{x}_i^r) = (\boldsymbol{w} \cdot \phi(x_i^r) + b) + (\boldsymbol{v}_r \cdot \phi_r(x_i^r) + d_r).$$

  There is a range of μ, which is unknown, in which the models combine a common and task-specific part.

Observe that (4.3) is a convex problem. As in the standard case of the L1-SVM, the corresponding dual problem is solved. To obtain the dual problem, it is necessary to express the Lagrangian of problem (4.3):

$$\begin{aligned}
&\mathcal{L}(\boldsymbol{w}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_T, b, d_1, \ldots, d_T, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\
&= C \sum_{r=1}^{T} \sum_{i=1}^{m_r} \xi_i^r + \frac{1}{2} \sum_{r=1}^{T} \|\boldsymbol{v}_r\|^2 + \frac{\mu}{2} \|\boldsymbol{w}\|^2 \\
&\quad - \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \left[\boldsymbol{w} \cdot \phi(x_i^r) + b + \boldsymbol{v}_r \cdot \phi_r(x_i^r) + d_r \right] - p_i^r + \xi_i^r \right\} \\
&\quad - \sum_{r=1}^{T} \sum_{i=1}^{m_r} \beta_i^r \xi_i^r,
\end{aligned} \tag{4.4}$$

where $\alpha_i^r, \beta_i^r \geq 0$ are the Lagrange multipliers. Here $\boldsymbol{\xi}$ represents the vector

$$(\xi_1^1, \ldots, \xi_{m_1}^1, \ldots, \xi_1^T, \ldots, \xi_{m_T}^T)^\intercal$$

and analogously we define $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Recall that the dual objective function is defined as

$$\begin{aligned}
\Theta(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \min_{\boldsymbol{w}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_T, b, d_1, \ldots, d_T, \boldsymbol{\xi}} \mathcal{L}(\boldsymbol{w}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_T, b, d_1, \ldots, d_T, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\
&= \mathcal{L}(\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta})
\end{aligned}$$

Since $\mathcal{L}$ is convex with respect to the primal variables, it is just necessary to compute the corresponding gradients

$$\nabla_{\boldsymbol{w}} \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies \mu \boldsymbol{w}^* - \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(x_i^r) \right\} = 0, \tag{4.5}$$

$$\nabla_{\boldsymbol{v}_r} \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies \boldsymbol{v}_r{}^* - \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(x_i^r) \right\} = 0, \tag{4.6}$$

$$\nabla_b \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r y_i^r = 0, \tag{4.7}$$

$$\nabla_{\boldsymbol{v}_r} \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies \sum_{i=1}^{m_r} \alpha_i^r y_i^r = 0, \tag{4.8}$$

$$\nabla_{\xi_i^r} \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies C - \alpha_i^r - \beta_i^r = 0 \tag{4.9}$$

Using these results and substituting back in the Lagrangian we obtain

$$
\begin{aligned}
\Theta(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \mathcal{L}(\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\
&= \frac{1}{2} \sum_{r=1}^{T} \left\| \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(\boldsymbol{x}_i^r) \right\} \right\|^2 + \frac{\mu}{2} \left\| \frac{1}{\mu} \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(\boldsymbol{x}_i^r) \right\} \right\|^2 \\
&\quad - \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \left[\left(\frac{1}{\mu} \sum_{s=1}^{T} \sum_{j=1}^{m_s} \alpha_j^s \left\{ y_j^s \phi(\boldsymbol{x}_j^s) \right\} \right) \cdot \phi(\boldsymbol{x}_i^r) \right] \right\} \\
&\quad - \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \left[\left(\sum_{j=1}^{m_r} \alpha_j^r \left\{ y_j^r \phi_r(\boldsymbol{x}_j^r) \right\} \right) \cdot \phi_r(\boldsymbol{x}_i^r) \right] \right\} \\
&\quad - \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ -p_i^r \right\} \\
&= \frac{1}{2} \sum_{r=1}^{T} \left\langle \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(\boldsymbol{x}_i^r) \right\}, \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(\boldsymbol{x}_i^r) \right\} \right\rangle \\
&\quad + \frac{\mu}{2} \left\langle \frac{1}{\mu} \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(\boldsymbol{x}_i^r) \right\}, \frac{1}{\mu} \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(\boldsymbol{x}_i^r) \right\} \right\rangle \\
&\quad - \frac{1}{\mu} \left\langle \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(\boldsymbol{x}_i^r) \right\}, \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(\boldsymbol{x}_i^r) \right\} \right\rangle \\
&\quad - \sum_{r=1}^{T} \left\langle \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(\boldsymbol{x}_i^r) \right\}, \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(\boldsymbol{x}_i^r) \right\} \right\rangle \\
&\quad - \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ -p_i^r \right\} \\
&= -\frac{1}{2\mu} \left\langle \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(\boldsymbol{x}_i^r) \right\}, \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(\boldsymbol{x}_i^r) \right\} \right\rangle \\
&\quad - \frac{1}{2} \sum_{r=1}^{T} \left\langle \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(\boldsymbol{x}_i^r) \right\}, \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(\boldsymbol{x}_i^r) \right\} \right\rangle \\
&\quad + \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r p_i^r
\end{aligned}
$$

Recall that the dual problem is defined as $\max_{\boldsymbol{\alpha}} \Theta(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha}$ fulfill the KKT conditions. The condition (4.9), using the $\alpha_i^r, \beta_i^r \geq 0$, implies $0 \leq \alpha_i^r \leq C$. Also, observe that the equality constraint (4.8) is included in the task-specific equality constraints (4.7). Taking

into account these KKT conditions, the dual problem can be expressed as

$$\begin{aligned} \min_{\alpha} \quad & \Theta(\alpha) = \frac{1}{2\mu} \left\langle \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(\boldsymbol{x}_i^r) \right\}, \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(\boldsymbol{x}_i^r) \right\} \right\rangle \\ & \frac{1}{2} \sum_{r=1}^{T} \left\langle \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(\boldsymbol{x}_i^r) \right\}, \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(\boldsymbol{x}_i^r) \right\} \right\rangle - \\ \text{s.t.} \quad & 0 \leq \alpha_i^r \leq C;\ i = 1, \ldots, m_r; r = 1, \ldots, T \\ & \sum_{i=1}^{m_r} = \alpha_i^r y_i^r; r = 1, \ldots, T. \end{aligned} \tag{4.10}$$

Using the kernel trick, we can write the dual problem using a vector formulation

$$\begin{aligned} \min_{\alpha} \quad & \Theta(\alpha) = \frac{1}{2} \boldsymbol{\alpha}^{\intercal} \left(\frac{1}{\mu} Q + K \right) \boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha} \\ \text{s.t.} \quad & 0 \leq \alpha_i^r \leq C_r;\ i = 1, \ldots, m_r; r = 1, \ldots, T \\ & \sum_{i=1}^{m_r} = \alpha_i^r y_i^r; r = 1, \ldots, T. \end{aligned} \tag{4.11}$$

Here Q and K are the common and specific kernel matrices, respectively. The matrix Q is generated using the common kernel, defined as

$$k(x_i^r, x_j^s) = \left\langle \phi(x_i^r), \phi(x_j^s) \right\rangle$$

and K is the block-diagonal matrix built using the kernel

$$k_r(x_i^r, x_j^s) = \delta_{rs} \left\langle \phi_r(x_i^r), \phi_s(x_j^s) \right\rangle$$

that is

$$Q = \begin{pmatrix} \underbrace{Q_{1,1}}_{m_1 \times m_1} & \underbrace{Q_{1,2}}_{m_1 \times m_2} & \underbrace{Q_{1,3}}_{m_1 \times m_3} & \cdots & \underbrace{Q_{1,T}}_{m_1 \times m_T} \\ \underbrace{Q_{2,1}}_{m_2 \times m_1} & \underbrace{Q_{2,2}}_{m_2 \times m_2} & \underbrace{Q_{2,1}}_{m_2 \times m_1} & \cdots & \underbrace{Q_{2,T}}_{m_2 \times m_T} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \underbrace{Q_{T,1}}_{m_T \times m_1} & \underbrace{Q_{T,2}}_{m_T \times m_2} & \underbrace{Q_{T,3}}_{m_T \times m_3} & \cdots & \underbrace{Q_{T,T}}_{m_T \times m_T} \end{pmatrix},$$

where each block $Q_{r,s}$

$$Q_{r,s} = \begin{pmatrix} k(\boldsymbol{x}_1^r, \boldsymbol{x}_1^s) & k(\boldsymbol{x}_1^r, \boldsymbol{x}_2^s) & \cdots & k(\boldsymbol{x}_1^r, \boldsymbol{x}_{m_s}^s) \\ k(\boldsymbol{x}_2^r, \boldsymbol{x}_1^s) & k(\boldsymbol{x}_2^r, \boldsymbol{x}_2^s) & \cdots & k(\boldsymbol{x}_2^r, \boldsymbol{x}_{m_s}^s) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_{m_r}^r, \boldsymbol{x}_1^s) & k(\boldsymbol{x}_{m_r}^r, \boldsymbol{x}_2^s) & \cdots & k(\boldsymbol{x}_{m_r}^r, \boldsymbol{x}_{m_s}^s) \end{pmatrix}; \tag{4.12}$$

and

$$K = \begin{pmatrix} \underbrace{K_{1,1}}_{m_1 \times m_1} & \underbrace{0}_{m_1 \times m_2} & \underbrace{0}_{m_1 \times m_3} & \cdots & \underbrace{0}_{m_1 \times m_T} \\ \underbrace{0}_{m_2 \times m_1} & \underbrace{K_{2,2}}_{m_2 \times m_2} & \underbrace{0}_{m_2 \times m_1} & \cdots & \underbrace{0}_{m_2 \times m_T} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \underbrace{0}_{m_T \times m_1} & \underbrace{0}_{m_T \times m_2} & \underbrace{0}_{m_T \times m_3} & \cdots & \underbrace{K_{T,T}}_{m_T \times T} \end{pmatrix},$$

where for each task block K_r we have

$$K_{r,r} = \begin{pmatrix} k_r(\boldsymbol{x}_1^r, \boldsymbol{x}_1^r) & k_r(\boldsymbol{x}_1^r, \boldsymbol{x}_2^r) & \cdots & k_r(\boldsymbol{x}_1^r, \boldsymbol{x}_{m_r}^r) \\ k_r(\boldsymbol{x}_2^r, \boldsymbol{x}_1^r) & k_r(\boldsymbol{x}_2^r, \boldsymbol{x}_2^r) & \cdots & k_r(\boldsymbol{x}_2^r, \boldsymbol{x}_{m_r}^r) \\ \vdots & \vdots & \ddots & \vdots \\ k_r(\boldsymbol{x}_{m_r}^r, \boldsymbol{x}_1^r) & k_r(\boldsymbol{x}_{m_r}^r, \boldsymbol{x}_2^r) & \cdots & k_r(\boldsymbol{x}_{m_r}^r, \boldsymbol{x}_{m_r}^r) \end{pmatrix}. \tag{4.13}$$

Combined, we have a multi-task kernel matrix $\widehat{Q} = (1/\mu)Q + K$, whose corresponding multi-task kernel function can be expressed as

$$\widehat{k}(\boldsymbol{x}_i^r, \boldsymbol{x}_j^s) = \frac{1}{\mu} k(\boldsymbol{x}_i^r, \boldsymbol{x}_j^s) + \delta_{rs} k_r(\boldsymbol{x}_i^r, \boldsymbol{x}_j^s).$$

The dual problem (4.22) is very similar to the standard one but we have two major differences: the use of the multi-task kernel matrix $\widehat{Q}$ and the multiple equality constraints. These constraints, which appear in (4.7) are consequence of the specific biases used in the primal problem (4.3). In Cai and Cherkassky (2012) the authors develop a Generalized SMO algorithm to account for these multiple equality constraints. Hyperparameter C_r is an upper bound for the dual coefficients, as in the standard case, but with a different bound for each task. The hyperparameter of interest for this MTL formulation is μ, which, in the dual problem, scales the common matrix Q. As with the primal formulation, we can define three different cases:

- Reduction to an ITL approach:

$$\mu \to \infty \implies \Theta(\alpha) = \frac{1}{2}\boldsymbol{\alpha}^\intercal (K)\, \boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha}.$$

  That is the matrix is block-diagonal and optimizing the dual problem is equivalent to optimizing a specific dual problem for each task.

- Reduction to a CTL approach:

$$C \to 0, \mu \to 0 \implies \Theta(\alpha) = \frac{1}{2}\boldsymbol{\alpha}^\intercal \left(\frac{1}{\mu} Q\right) \boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha}.$$

  That is the dual objective function is the standard one for common task learning.

- Pure MTL approach:

$$\mu_{\text{inf}} < \mu < \mu_{\text{sup}} \implies \Theta(\alpha) = \frac{1}{2}\boldsymbol{\alpha}^\intercal \left(\frac{1}{\mu} Q + K\right) \boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha}.$$

There is a range of μ, which is unknown, in which the kernel matrix combines the common and specific matrices.

4.2.1.2 *convex* MTL L1-SVM

The *convex* MTL primal problem formulation, presented in Ruiz et al. (2019), changes the formulation of the *additive* MTL SVM but changes its formulation for a convex one that is more interpretable one. The primal problem is

$$\begin{aligned} \underset{w,\boldsymbol{v}_r,\xi}{\arg\min} \quad & J(\boldsymbol{w},\boldsymbol{v}_r,\xi) = C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\xi_i^r + \frac{1}{2}\sum_{r=1}^{T}\|\boldsymbol{v}_r\|^2 + \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \quad & y_i^r\left(\lambda_r\left\{\boldsymbol{w}\cdot\phi(x_i^r)+b\right\} + (1-\lambda_r)\left\{\boldsymbol{v}_r\cdot\phi_r(x_i^r)+d_r\right\}\right) \geq p_i^r - \xi_i^r, \\ & \xi_i^r \geq 0;\ i=1,\ldots,m_r,\ r=1,\ldots,T. \end{aligned} \tag{4.14}$$

Here, the former hyperparameter μ used in the regularization is replaced by λ_r, which is used in the model definition. The prediction model is

$$h_r(\boldsymbol{x}) = \lambda_r\left\{\boldsymbol{w}\cdot\phi(x_i^r)+b\right\} + (1-\lambda_r)\left\{\boldsymbol{v}_r\cdot\phi_r(x_i^r)+d_r\right\}$$

for regression and

$$h_r(\boldsymbol{x}) = \operatorname{sign}\left(\lambda_r\left\{\boldsymbol{w}\cdot\phi(x_i^r)+b\right\} + (1-\lambda_r)\left\{\boldsymbol{v}_r\cdot\phi_r(x_i^r)+d_r\right\}\right)$$

for classification.

With this convex formulation, the roles of hyperparameters C and λ_r are independent. Hyperparameter C regulates the trade-off between the loss and the margin size of each task-specialized model h_r, while λ_r indicates how specific or common these models are in the range of $[0,1]$. With $\lambda_r = 0$ we have independent models for each task and for $\lambda_r = 1$ we have a common model for all tasks. The cases described for the *additive* approach are now:

In (4.3) there are two hyperparameters: C and μ, which, in combination, balance the different parts of the objective function.

- Reduction to an ITL approach:

$$\lambda_r = 0; r = 1,\ldots,T \implies h_r(\boldsymbol{x}_i^r) = \boldsymbol{v}_r\cdot\phi_r(x_i^r) + d_r.$$

- Reduction to a CTL approach:

$$\lambda_r = 1; r = 1,\ldots,T \implies h_r(\boldsymbol{x}_i^r) = \boldsymbol{w}\cdot\phi(x_i^r) + b.$$

- Pure MTL approach:

$$0 < \lambda_r < 1; r = 1,\ldots,T \implies h_r(\boldsymbol{x}_i^r) = \lambda_r(\boldsymbol{w}\cdot\phi(x_i^r)+b) + (1-\lambda_r)(\boldsymbol{v}_r\cdot\phi_r(x_i^r)+d_r).$$

Observe that now the cases are not asymptotical but have clear values, 0 for ITL and 1 for MTL, while all the values in the open $(0,1)$ yield pure MTL models. Also, the parameter C no longer interferes with these cases and only λ_r calibrates the specifity of

the models. To obtain the dual problem the Lagrangian of problem (4.14):

$$\begin{aligned}
&\mathcal{L}(\boldsymbol{w}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_T, b, d_1, \ldots, d_T, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\
&= C \sum_{r=1}^{T} \sum_{i=1}^{m_r} \xi_i^r + \frac{1}{2} \sum_{r=1}^{T} \|\boldsymbol{v}_r\|^2 + \frac{1}{2} \|\boldsymbol{w}\|^2 \\
&\quad - \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \left(\lambda_r \left\{ \boldsymbol{w} \cdot \phi(x_i^r) + b \right\} + (1 - \lambda_r) \left\{ \boldsymbol{v}_r \cdot \phi_r(x_i^r) + d_r \right\} \right) - p_i^r + \xi_i^r \right\} \\
&\quad - \sum_{r=1}^{T} \sum_{i=1}^{m_r} \beta_i^r \xi_i^r,
\end{aligned} \tag{4.15}$$

where $\alpha_i^r, \beta_i^r \geq 0$ are the Lagrange multipliers. Again, $\boldsymbol{\xi}$ represents the vector

$$(\xi_1^1, \ldots, \xi_{m_1}^1, \ldots, \xi_1^T, \ldots, \xi_{m_T}^T)^\intercal$$

and analogously for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.The dual objective function is defined as

$$\begin{aligned}
\Theta(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \min_{\boldsymbol{w}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_T, b, d_1, \ldots, d_T, \boldsymbol{\xi}} \mathcal{L}(\boldsymbol{w}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_T, b, d_1, \ldots, d_T, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\
&= \mathcal{L}(\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta})
\end{aligned}$$

The gradients with respect to the primal variables are

$$\nabla_{\boldsymbol{w}} \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies \boldsymbol{w}^* - \lambda_r \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi(x_i^r) \right\} = 0, \tag{4.16}$$

$$\nabla_{\boldsymbol{v}_r} \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies \boldsymbol{v}_r{}^* - (1 - \lambda_r) \sum_{i=1}^{m_r} \alpha_i^r \left\{ y_i^r \phi_r(x_i^r) \right\} = 0, \tag{4.17}$$

$$\nabla_b \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r y_i^r = 0, \tag{4.18}$$

$$\nabla_{\boldsymbol{v}_r} \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies \sum_{i=1}^{m_r} \alpha_i^r y_i^r = 0, \tag{4.19}$$

$$\nabla_{\xi_i^r} \mathcal{L}|_{\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}} = 0 \implies C - \alpha_i^r - \beta_i^r = 0 \tag{4.20}$$

Using these results and substituting back in the Lagrangian we obtain

$$
\begin{aligned}
&\Theta(\boldsymbol{\alpha},\boldsymbol{\beta}) = \mathcal{L}(\boldsymbol{w}^*, \boldsymbol{v}_1{}^*, \ldots, \boldsymbol{v}_T{}^*, b^*, d_1{}^*, \ldots, d_T{}^*, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\
&= \frac{1}{2}\sum_{r=1}^{T}\left\|(1-\lambda_r)\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi_r(\boldsymbol{x}_i^r)\right\}\right\|^2 + \frac{1}{2}\left\|\sum_{r=1}^{T}\lambda_r\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi(\boldsymbol{x}_i^r)\right\}\right\|^2 \\
&\quad - \sum_{r=1}^{T}\lambda_r\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\left[\left(\sum_{s=1}^{T}\lambda_r\sum_{j=1}^{m_s}\alpha_j^s\left\{y_j^s\phi(\boldsymbol{x}_j^s)\right\}\right)\cdot\phi(\boldsymbol{x}_i^r)\right]\right\} \\
&\quad - \sum_{r=1}^{T}(1-\lambda_r)\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\left[\left((1-\lambda_r)\sum_{j=1}^{m_r}\alpha_j^r\left\{y_j^r\phi_r(\boldsymbol{x}_j^r)\right\}\right)\cdot\phi_r(\boldsymbol{x}_i^r)\right]\right\} \\
&\quad - \sum_{r=1}^{T}\sum_{i=1}^{m_r}\alpha_i^r\left\{-p_i^r\right\} \\
&= \frac{1}{2}\sum_{r=1}^{T}\left\langle(1-\lambda_r)\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi_r(\boldsymbol{x}_i^r)\right\}, (1-\lambda_r)\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi_r(\boldsymbol{x}_i^r)\right\}\right\rangle \\
&\quad + \frac{1}{2}\left\langle\lambda_r\sum_{r=1}^{T}\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi(\boldsymbol{x}_i^r)\right\}, \lambda_r\sum_{r=1}^{T}\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi(\boldsymbol{x}_i^r)\right\}\right\rangle \\
&\quad - \left\langle\sum_{r=1}^{T}\lambda_r\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi(\boldsymbol{x}_i^r)\right\}, \sum_{r=1}^{T}\lambda_r\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi(\boldsymbol{x}_i^r)\right\}\right\rangle \\
&\quad - \sum_{r=1}^{T}\left\langle(1-\lambda_r)\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi_r(\boldsymbol{x}_i^r)\right\}, (1-\lambda_r)\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi_r(\boldsymbol{x}_i^r)\right\}\right\rangle \\
&\quad - \sum_{r=1}^{T}\sum_{i=1}^{m_r}\alpha_i^r\left\{-p_i^r\right\} \\
&= -\frac{1}{2}\left\langle\sum_{r=1}^{T}\lambda_r\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi(\boldsymbol{x}_i^r)\right\}, \sum_{r=1}^{T}\lambda_r\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi(\boldsymbol{x}_i^r)\right\}\right\rangle \\
&\quad - \frac{1}{2}\sum_{r=1}^{T}(1-\lambda_r)\left\langle\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi_r(\boldsymbol{x}_i^r)\right\}, (1-\lambda_r)\sum_{i=1}^{m_r}\alpha_i^r\left\{y_i^r\phi_r(\boldsymbol{x}_i^r)\right\}\right\rangle \\
&\quad + \sum_{r=1}^{T}\sum_{i=1}^{m_r}\alpha_i^r p_i^r
\end{aligned}
$$

Recall that the dual problem is defined as $\max_{\boldsymbol{\alpha}}\Theta(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha}$ fulfill the KKT conditions. The condition (4.9), using the $\alpha_i^r, \beta_i^r \geq 0$, implies $0 \leq \alpha_i^r \leq C$. Also, observe that the equality constraint (4.8) is included in the task-specific equality constraints (4.7). Taking

into account these KKT conditions, the dual problem can be expressed as

$$\begin{aligned}\min_{\alpha} \quad & \Theta(\alpha) = \frac{1}{2}\left\langle \lambda_r \sum_{r=1}^{T}\sum_{i=1}^{m_r} \alpha_i^r \left\{y_i^r \phi(\boldsymbol{x}_i^r)\right\}, \lambda_r \sum_{r=1}^{T}\sum_{i=1}^{m_r} \alpha_i^r \left\{y_i^r \phi(\boldsymbol{x}_i^r)\right\} \right\rangle \\ & \frac{1}{2}\sum_{r=1}^{T}\left\langle (1-\lambda_r)\sum_{i=1}^{m_r} \alpha_i^r \left\{y_i^r \phi_r(\boldsymbol{x}_i^r)\right\}, (1-\lambda_r)\sum_{i=1}^{m_r} \alpha_i^r \left\{y_i^r \phi_r(\boldsymbol{x}_i^r)\right\} \right\rangle - \\ \text{s.t.} \quad & 0 \leq \alpha_i^r \leq C;\ i = 1, \ldots, m_r; r = 1, \ldots, T \\ & \sum_{i=1}^{m_r} = \alpha_i^r y_i^r; r = 1, \ldots, T.\end{aligned} \tag{4.21}$$

Using the kernel trick, we can write the dual problem using a vector formulation

$$\begin{aligned}\min_{\alpha} \quad & \Theta(\alpha) = \frac{1}{2}\boldsymbol{\alpha}^\intercal \left(\Lambda Q \Lambda + (I_N - \Lambda) K (I_N - \Lambda)\right)\boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha} \\ \text{s.t.} \quad & 0 \leq \alpha_i^r \leq C;\ i = 1, \ldots, m_r; r = 1, \ldots, T \\ & \sum_{i=1}^{m_r} = \alpha_i^r y_i^r; r = 1, \ldots, T.\end{aligned} \tag{4.22}$$

where

$$\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_T)$$

and I_N. Here Q and K are the common and specific kernel matrices, respectively. Combined, we have a multi-task kernel matrix $\widehat{Q} = \Lambda Q \Lambda + (I_N - \Lambda) K (I_N - \Lambda)$, whose corresponding multi-task kernel function can be expressed as

$$\widehat{k}(\boldsymbol{x}_i^r, \boldsymbol{x}_j^s) = \lambda_r^2 k(\boldsymbol{x}_i^r, \boldsymbol{x}_j^s) + \delta_{rs}(1-\lambda_r)^2 k_r(\boldsymbol{x}_i^r, \boldsymbol{x}_j^s).$$

This dual problem is very similar to the one shown in (4.22) where there are also T equality constraints, but the multi-task kernel matrix $\widehat{Q}$ is defined differently, dropping the μ hyperparameter and incorporating the λ_r ones. Studying the influence of λ_r hyperparameters in the dual problem we can describe the following cases:

- Reduction to an ITL approach:

$$\lambda_r = 0; r = 1, \ldots, T \implies \Theta(\alpha) = \frac{1}{2}\boldsymbol{\alpha}^\intercal (K)\boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha}.$$

- Reduction to a CTL approach:

$$\lambda_r = 1; r = 1, \ldots, T \implies \Theta(\alpha) = \frac{1}{2}\boldsymbol{\alpha}^\intercal (Q)\boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha}.$$

- Pure MTL approach:

$$0 < \lambda_r < 1; r = 1, \ldots, T \implies \Theta(\alpha) = \frac{1}{2}\boldsymbol{\alpha}^\intercal \left(\Lambda Q \Lambda + (I_N - \Lambda) K (I_N - \Lambda)\right)\boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha}.$$

The properties that are found in the primal formulation are also present in the dual one. All the values of λ_r in the open interval $(0, 1)$ correspond to pure MTL approaches while the extreme values $\lambda_r = 1$ and $\lambda_r = 0$ correspond to CTL and ITL approaches, respectively.

Both the *additive* and *convex* MTL SVM approaches solve a similar problem, but there is a change in the formulation to get rid of a regularization hyperparameter μ in favor of those defining convex combination of models, hyperparameters λ_r. Both approaches offer similar properties: ranging the value of their hyperparameters to go from completely common to completely independent models, and passing through pure multi-task models. However, it is not totally trivial what is the relation between those two approaches. In Ruiz et al. (2019) we provide two propositions to show the equivalence between *additive* and *convex* MTL SVM formulations.

Proposition 1. The *additive* MTL-SVM primal problem with parameters C_{add} and μ (and possibly ϵ) and the *convex* MTL-SVM primal problem with parameters C_{conv} and $\lambda_1 = \ldots = \lambda_T = \lambda$ (and possibly ϵ), with $\lambda \in (0, 1)$, are equivalent when $C_{\text{add}} = (1-\lambda)^2 C_{\text{conv}}$ and $\mu = (1-\lambda)^2/\lambda^2$.

Proof. Making the change of variables $w = \lambda u$, $v_r = (1-\lambda)u_r$, $b = \lambda\hat{b}$ and $d_r = (1-\lambda)\hat{d}_r$ in the convex primal problem, we can write it as

$$
\begin{aligned}
\underset{w,v_r,\xi}{\arg\min} \quad & J(w, v_r, \xi) = C_{\text{conv}} \sum_{t=1}^{T} \sum_{i=1}^{m_r} \xi_i^r + \frac{1}{2(1-\lambda)^2} \sum_{t=1}^{T} \|v_r\|^2 + \frac{1}{2\lambda^2}\|w\|^2 \\
\text{s.t.} \quad & y_i^r (w \cdot \phi(x_i^r) + b + v_r \cdot \phi_r(x_i^r) + d_r) \geq p_i^r - \xi_i^r, \\
& \xi_i^r \geq 0, \\
& i = 1, \ldots, m_r, \; t = 1, \ldots, T.
\end{aligned}
$$

Multiplying now the objective function by $(1-\lambda)^2$ we obtain the additive MTL-SVM primal problem with $\mu = (1-\lambda)^2/\lambda^2$ and $C_{\text{add}} = (1-\lambda)^2 C_{\text{conv}}$. Conversely, we can start at the primal additive problem and make the inverse changes to arrive now to the primal convex problem. □ □

4.2.2 L2 Support Vector Machine

4.2.3 LS Support Vector Machine

4.3 Optimal Convex Combination of fitted Models

4.4 Convex Multi-Task Learning with Neural Networks

4.5 Experiments

4.6 Conclusions

In this chapter, we have...

Chapter 5

Adaptive Graph Laplacian Multi-Task Support Vector Machine

5.1 Introduction

5.2 Graph Laplacian Multi-Task Support Vector Machine

In **?** we proposed a convex formulation of the Graph Laplacian MTL SVM which includes a common regularization term and whose primal problem is

$$\begin{aligned} \underset{\boldsymbol{w},\boldsymbol{v}_1,\ldots,\boldsymbol{v}_T,b,\boldsymbol{\xi}}{\arg\min} \quad & \sum_{r=1}^{T} C_r \sum_{i=1}^{n_r} \xi_i^r + \frac{\nu}{2} \sum_{r=1}^{T} \sum_{s=1}^{T} A_{rs} \|\boldsymbol{v}_r - \boldsymbol{v}_s\|^2 + \frac{1}{2} \sum_r \|\boldsymbol{v}_r\|^2 + \frac{1}{2} \|\boldsymbol{w}\|^2 \\ \text{s.t.} \quad & y_i^r(\lambda(\boldsymbol{w} \cdot \boldsymbol{x}_i^r) + (1-\lambda)(\boldsymbol{v}_r \cdot \boldsymbol{x}_i^r) + b_r) \geq p_i^r - \xi_i^r, \\ & \xi_i^r \geq 0,\ i = 1,\ldots,n_r,\ r = 1,\ldots,T. \end{aligned} \tag{5.1}$$

The corresponding Lagrangian is

$$\begin{aligned} &\mathcal{L}(\boldsymbol{w}, \boldsymbol{v}_r, b_r, \xi_i^r, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\ &= \sum_{r=1}^{T} C_r \sum_{i=1}^{n_r} \xi_i^r + \frac{\nu}{2} \sum_{r=1}^{T} \sum_{s=1}^{T} A_{rs} \|\boldsymbol{v}_r - \boldsymbol{v}_s\|^2 + \frac{1}{2} \sum_r \|\boldsymbol{v}_r\|^2 + \frac{1}{2} \|\boldsymbol{w}\|^2 \\ &\quad - \sum_{r=1}^{T} \sum_{i=1}^{n_r} \alpha_i^r [y_i^r(\lambda(\boldsymbol{w} \cdot \boldsymbol{x}_i^r) + (1-\lambda)(\boldsymbol{v}_r \cdot \boldsymbol{x}_i^r) + b_r) - p_i^r + \xi_i^r] - \sum_{r=1}^{T} \sum_{i=1}^{n_r} \beta_i^r \xi_i^r. \end{aligned} \tag{5.2}$$

Taking derivatives

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = 0 &\implies \boldsymbol{w} = \lambda \sum_{r=1}^{T} \sum_{i=1}^{m_r} \alpha_i^r y_i^r x_i^r \ , \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{v}_r} = 0 &\implies \boldsymbol{v}_r + \sum_{s=1}^{T} (L_{rs} + L_{sr})(\boldsymbol{v}_r^* - \boldsymbol{v}_s^*) = \sum_{i=1}^{m_r} \alpha_i^r y_i^r x_i^r \ , \\
\frac{\partial \mathcal{L}}{\partial b_r} = 0 &\implies \sum_{i=1}^{m_r} \alpha_i^r y_i^r = 0 \ , \\
\frac{\partial \mathcal{L}}{\partial \xi_i^r} = 0 &\implies C_r - \alpha_i^r - \beta_i^r = 0 \ .
\end{aligned}$$

Observe that

$$\begin{aligned}
\boldsymbol{v}^\intercal (I_T \otimes I_d) \boldsymbol{v} &= \sum_{r=1}^{T} \|v_r\|^2 \, , \\
\boldsymbol{v}^\intercal (L \otimes I_d) \boldsymbol{v} &= \frac{1}{2} \sum_{r=1}^{T} \sum_{s=1}^{T} A_{rs} \|v_r - v_s\|^2 \, ,
\end{aligned}$$

Proof:

$$\begin{aligned}
\boldsymbol{v}(L \otimes I_d)\boldsymbol{v} &= \boldsymbol{v}(D \otimes I_d)\boldsymbol{v} - \boldsymbol{v}(A \otimes I_d)\boldsymbol{v} \\
&= \sum_{r=1}^{T} \sum_{s=1}^{T} D_{rs} v_r^\intercal v_s - \sum_{r=1}^{T} \sum_{s=1}^{T} A_{rs} v_r^\intercal v_s \\
&= \sum_{r=1}^{T} D_{rr} v_r^\intercal v_r - \sum_{r=1}^{T} \sum_{s=1}^{T} A_{rs} v_r^\intercal v_s \\
&= \sum_{r=1}^{T} \sum_{s=1}^{T} A_{rs} v_r^\intercal v_r - \sum_{r=1}^{T} \sum_{s=1}^{T} A_{rs} v_r^\intercal v_s \\
&= \sum_{r=1}^{T} \sum_{s=1}^{T} A_{rs} (v_r^\intercal v_r - v_r^\intercal v_s)
\end{aligned}$$

If A is symmetric, that is $A_{rs} = A_{sr}$, then

$$\begin{aligned}
\sum_{r=1}^{T}\sum_{s=1}^{T} A_{rs}(v_r^\intercal v_r - v_r^\intercal v_s) &= \sum_{r=1}^{T}\sum_{s=r}^{T}\{A_{rs}(v_r^\intercal v_r - v_r^\intercal v_s) + A_{sr}(v_s^\intercal v_s - v_s^\intercal v_r)\} \\
&= \sum_{r=1}^{T}\sum_{s=r}^{T}\{(A_{rs} + A_{sr})(v_r^\intercal v_r + v_s^\intercal v_s - 2v_r^\intercal v_s)\} \\
&= \sum_{r=1}^{T}\sum_{s=r}^{T}\{(A_{rs} + A_{sr})\,\|w_r - w_s\|^2\} \\
&= \frac{1}{2}\sum_{r=1}^{T}\sum_{s=1}^{T}\{(A_{rs} + A_{sr})\,\|w_r - w_s\|^2\} \\
&= \sum_{r=1}^{T}\sum_{s=1}^{T}\{A_{rs}\,\|w_r - w_s\|^2\}.
\end{aligned}$$

5.3 Adaptive Graph Laplacian Algorithm

If we want to learn A from data we would like the matrix to meet some requirements:

- A has to be symmetric, so we can express the regularizer using the Laplacian in the dual form.
- The rows of A add up to 1.

5.4 Experiments

5.5 Conclusions

In this chapter, we have...

Bibliography

Agarwal, A., III, H. D., and Gerber, S. (2010). Learning multiple tasks using manifold regularization. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A., editors, *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 46–54. Curran Associates, Inc.

Akhiezer, N. I. and Glazman, I. M. (1961). Theory of linear operators in hilbert space.

Álvarez, M. A., Rosasco, L., and Lawrence, N. D. (2012). Kernels for vector-valued functions: A review. *Found. Trends Mach. Learn.*, 4(3):195–266.

Ando, R. K. and Zhang, T. (2005). A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, 6:1817–1853.

Argyriou, A., Clémençon, S., and Zhang, R. (2013). Learning the graph of relations among multiple tasks. *HAL*.

Argyriou, A., Evgeniou, T., and Pontil, M. (2006). Multi-task feature learning. In Schölkopf, B., Platt, J. C., and Hofmann, T., editors, *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pages 41–48. MIT Press.

Argyriou, A., Evgeniou, T., and Pontil, M. (2008). Convex multi-task feature learning. *Mach. Learn.*, 73(3):243–272.

Argyriou, A., Micchelli, C. A., Pontil, M., and Ying, Y. (2007). A spectral regularization framework for multi-task structure learning. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T., editors, *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 25–32. Curran Associates, Inc.

Baldassarre, L., Rosasco, L., Barla, A., and Verri, A. (2012). Multi-output learning via spectral filtering. *Mach. Learn.*, 87(3):259–301.

Bartlett, P. L. and Mendelson, S. (2002). Rademacher and gaussian complexities: Risk bounds and structural results. *J. Mach. Learn. Res.*, 3:463–482.

Barzilai, A. and Crammer, K. (2015). Convex multi-task learning by clustering. In Lebanon, G. and Vishwanathan, S. V. N., editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, California, USA, May 9-12, 2015*, volume 38 of *JMLR Workshop and Conference Proceedings*. JMLR.org.

Baxter, J. (2000). A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198.

Ben-David, S. and Borbely, R. S. (2008). A notion of task relatedness yielding provable multiple-task learning guarantees. *Mach. Learn.*, 73(3):273–287.

Ben-David, S. and Schuller, R. (2003). Exploiting task relatedness for mulitple task learning. In Schölkopf, B. and Warmuth, M. K., editors, *Computational Learning Theory and Kernel Machines, 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003, Proceedings*, volume 2777 of *Lecture Notes in Computer Science*, pages 567–580. Springer.

Bonilla, E. V., Chai, K. M. A., and Williams, C. K. I. (2007). Multi-task gaussian process prediction. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T., editors, *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 153–160. Curran Associates, Inc.

Cai, F. and Cherkassky, V. (2009). SVM+ regression and multi-task learning. In *International Joint Conference on Neural Networks, IJCNN 2009, Atlanta, Georgia, USA, 14-19 June 2009*, pages 418–424. IEEE Computer Society.

Cai, F. and Cherkassky, V. (2012). Generalized SMO algorithm for svm-based multitask learning. *IEEE Trans. Neural Networks Learn. Syst.*, 23(6):997–1003.

Caruana, R. (1997). Multitask learning. *Mach. Learn.*, 28(1):41–75.

Cavallanti, G., Cesa-Bianchi, N., and Gentile, C. (2010). Linear algorithms for online multitask classification. *J. Mach. Learn. Res.*, 11:2901–2934.

Chen, J., Liu, J., and Ye, J. (2010). Learning incoherent sparse and low-rank patterns from multiple tasks. In Rao, B., Krishnapuram, B., Tomkins, A., and Yang, Q., editors, *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, pages 1179–1188. ACM.

Chen, J., Tang, L., Liu, J., and Ye, J. (2009). A convex formulation for learning shared structures from multiple tasks. In Danyluk, A. P., Bottou, L., and Littman, M. L., editors, *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 137–144. ACM.

Chen, J., Zhou, J., and Ye, J. (2011). Integrating low-rank and group-sparse structures for robust multi-task learning. In Apté, C., Ghosh, J., and Smyth, P., editors, *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011*, pages 42–50. ACM.

Ciliberto, C., Mroueh, Y., Poggio, T. A., and Rosasco, L. (2015). Convex learning of multiple tasks and their structure. In Bach, F. R. and Blei, D. M., editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1548–1557. JMLR.org.

Crammer, K. and Mansour, Y. (2012). Learning multiple tasks using shared hypotheses. In Bartlett, P. L., Pereira, F. C. N., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1484–1492.

Dinuzzo, F. (2013). Learning output kernels for multi-task problems. *Neurocomputing*, 118:119–126.

Evgeniou, T., Micchelli, C. A., and Pontil, M. (2005). Learning multiple tasks with kernel methods. *J. Mach. Learn. Res.*, 6:615–637.

Evgeniou, T. and Pontil, M. (2004). Regularized multi–task learning. In Kim, W., Kohavi, R., Gehrke, J., and DuMouchel, W., editors, *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 109–117. ACM.

Fung, G. and Mangasarian, O. L. (2001). Proximal support vector machine classifiers. In Lee, D., Schkolnick, M., Provost, F. J., and Srikant, R., editors, *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, San Francisco, CA, USA, August 26-29, 2001*, pages 77–86. ACM.

Gong, P., Ye, J., and Zhang, C. (2012a). Multi-stage multi-task feature learning. In Bartlett, P. L., Pereira, F. C. N., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1997–2005.

Gong, P., Ye, J., and Zhang, C. (2012b). Robust multi-task feature learning. In Yang, Q., Agarwal, D., and Pei, J., editors, *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, pages 895–903. ACM.

Han, L. and Zhang, Y. (2015). Learning tree structure in multi-task learning. In Cao, L., Zhang, C., Joachims, T., Webb, G. I., Margineantu, D. D., and Williams, G., editors, *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 397–406. ACM.

Han, L. and Zhang, Y. (2016). Multi-stage multi-task learning with reduced rank. In Schuurmans, D. and Wellman, M. P., editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 1638–1644. AAAI Press.

Hein, M., Bousquet, O., Hein, M., and Bousquet, O. (2004). Kernels, associated structures and generalizations.

Hernández-Lobato, D. and Hernández-Lobato, J. M. (2013). Learning feature selection dependencies in multi-task learning. In Burges, C. J. C., Bottou, L., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 746–754.

Hernández-Lobato, D., Hernández-Lobato, J. M., and Ghahramani, Z. (2015). A probabilistic model for dirty multi-task feature selection. In Bach, F. R. and Blei, D. M., editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1073–1082. JMLR.org.

III, H. D. (2007). Frustratingly easy domain adaptation. In Carroll, J. A., van den Bosch, A., and Zaenen, A., editors, *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. The Association for Computational Linguistics.

III, H. D. (2009). Bayesian multitask learning with latent hierarchies. In Bilmes, J. A. and Ng, A. Y., editors, *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pages 135–142. AUAI Press.

Jacob, L., Bach, F. R., and Vert, J. (2008). Clustered multi-task learning: A convex formulation. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, pages 745–752. Curran Associates, Inc.

Jalali, A., Ravikumar, P., Sanghavi, S., and Ruan, C. (2010). A dirty model for multi-task learning. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A., editors, *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 964–972. Curran Associates, Inc.

Jawanpuria, P. and Nath, J. S. (2012). A convex feature learning formulation for latent task structure discovery. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress.

Jebara, T. (2004). Multi-task feature and kernel selection for svms. In Brodley, C. E., editor, *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, volume 69 of *ACM International Conference Proceeding Series*. ACM.

Jebara, T. (2011). Multitask sparsity via maximum entropy discrimination. *J. Mach. Learn. Res.*, 12:75–110.

Jeong, J. and Jun, C. (2018). Variable selection and task grouping for multi-task learning. In Guo, Y. and Farooq, F., editors, *Proceedings of the 24th ACM SIGKDD*

International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018, pages 1589–1598. ACM.

Kadri, H., Duflos, E., Preux, P., Canu, S., Rakotomamonjy, A., and Audiffren, J. (2016). Operator-valued kernels for learning from functional response data. *J. Mach. Learn. Res.*, 17:20:1–20:54.

Kang, Z., Grauman, K., and Sha, F. (2011). Learning with whom to share in multi-task feature learning. In Getoor, L. and Scheffer, T., editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 521–528. Omnipress.

Kumar, A. and III, H. D. (2012). Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress.

Lawrence, N. D. and Platt, J. C. (2004). Learning to learn with the informative vector machine. In Brodley, C. E., editor, *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, volume 69 of *ACM International Conference Proceeding Series*. ACM.

Li, Y., Tian, X., Song, M., and Tao, D. (2015). Multi-task proximal support vector machine. *Pattern Recognit.*, 48(10):3249–3257.

Liang, L. and Cherkassky, V. (2008). Connection between SVM+ and multi-task learning. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN 2008, part of the IEEE World Congress on Computational Intelligence, WCCI 2008, Hong Kong, China, June 1-6, 2008*, pages 2048–2054. IEEE.

Liu, H., Palatucci, M., and Zhang, J. (2009). Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery. In Danyluk, A. P., Bottou, L., and Littman, M. L., editors, *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 649–656. ACM.

Long, M. and Wang, J. (2015). Learning multiple tasks with deep relationship networks. *CoRR*, abs/1506.02117.

Lozano, A. C. and Swirszcz, G. (2012). Multi-level lasso for sparse multi-task regression. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress.

Maurer, A. (2006a). Bounds for linear multi-task learning. *J. Mach. Learn. Res.*, 7:117–139.

Maurer, A. (2006b). The rademacher complexity of linear transformation classes. In Lugosi, G. and Simon, H. U., editors, *Learning Theory, 19th Annual Conference on Learning Theory, COLT 2006, Pittsburgh, PA, USA, June 22-25, 2006, Proceedings*, volume 4005 of *Lecture Notes in Computer Science*, pages 65–78. Springer.

Maurer, A. (2009). Transfer bounds for linear feature learning. *Mach. Learn.*, 75(3):327–350.

Maurer, A. and Pontil, M. (2010). K -dimensional coding schemes in hilbert spaces. *IEEE Trans. Inf. Theory*, 56(11):5839–5846.

Maurer, A., Pontil, M., and Romera-Paredes, B. (2013). Sparse coding for multitask and transfer learning. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 343–351. JMLR.org.

Maurer, A., Pontil, M., and Romera-Paredes, B. (2016). The benefit of multitask representation learning. *J. Mach. Learn. Res.*, 17:81:1–81:32.

Micchelli, C. A. and Pontil, M. (2004). Kernels for multi–task learning. In *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, pages 921–928.

Micchelli, C. A. and Pontil, M. (2005). On learning vector-valued functions. *Neural Comput.*, 17(1):177–204.

Misra, I., Shrivastava, A., Gupta, A., and Hebert, M. (2016). Cross-stitch networks for multi-task learning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 3994–4003. IEEE Computer Society.

Obozinski, G., Taskar, B., and Jordan, M. (2006). Multi-task feature selection. *Statistics Department, UC Berkeley, Tech. Rep*, 2(2.2):2.

Parameswaran, S. and Weinberger, K. Q. (2010). Large margin multi-task metric learning. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A., editors, *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 1867–1875. Curran Associates, Inc.

Pong, T. K., Tseng, P., Ji, S., and Ye, J. (2010). Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM J. Optim.*, 20(6):3465–3489.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098.

Ruder, S., Bingel, J., Augenstein, I., and Søgaard, A. (2017). Sluice networks: Learning what to share between loosely related tasks. *CoRR*, abs/1705.08142.

Ruiz, C., Alaíz, C. M., and Dorronsoro, J. R. (2019). A convex formulation of svm-based multi-task learning. In *HAIS 2019*, volume 11734 of *Lecture Notes in Computer Science*, pages 404–415. Springer.

Ruiz, C., Alaíz, C. M., and Dorronsoro, J. R. (2021). Convex formulation for multi-task l1-, l2-, and ls-svms. *Neurocomputing*, 456:599–608.

Sun, X., Panda, R., Feris, R., and Saenko, K. (2020). Adashare: Learning what to share for efficient deep multi-task learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Suykens, J. A. K. and Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural Process. Lett.*, 9(3):293–300.

Thrun, S. and O'Sullivan, J. (1996). Discovering structure in multiple learning tasks: The TC algorithm. In Saitta, L., editor, *Machine Learning, Proceedings of the Thirteenth International Conference (ICML '96), Bari, Italy, July 3-6, 1996*, pages 489–497. Morgan Kaufmann.

Vapnik, V. (1982). *Estimation of dependences based on empirical data*. Springer Science & Business Media.

Vapnik, V. (2000). *The Nature of Statistical Learning Theory*. Statistics for Engineering and Information Science. Springer.

Vapnik, V. and Izmailov, R. (2015). Learning using privileged information: similarity control and knowledge transfer. *J. Mach. Learn. Res.*, 16:2023–2049.

Vapnik, V. and Vashist, A. (2009). A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5-6):544–557.

Weinberger, K. Q. and Saul, L. K. (2009). Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244.

Xu, S., An, X., Qiao, X., and Zhu, L. (2014). Multi-task least-squares support vector machines. *Multim. Tools Appl.*, 71(2):699–715.

Xue, Y., Liao, X., Carin, L., and Krishnapuram, B. (2007). Multi-task learning for classification with dirichlet process priors. *J. Mach. Learn. Res.*, 8:35–63.

Yang, Y. and Hospedales, T. M. (2017a). Deep multi-task representation learning: A tensor factorisation approach. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Yang, Y. and Hospedales, T. M. (2017b). Trace norm regularised deep multi-task learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Yu, K., Tresp, V., and Schwaighofer, A. (2005). Learning gaussian processes from multiple tasks. In Raedt, L. D. and Wrobel, S., editors, *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, volume 119 of *ACM International Conference Proceeding Series*, pages 1012–1019. ACM.

Zhang, Y. and Yeung, D. (2010). A convex formulation for learning task relationships in multi-task learning. In Grünwald, P. and Spirtes, P., editors, *UAI 2010, Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, Catalina Island, CA, USA, July 8-11, 2010*, pages 733–442. AUAI Press.

Zhang, Y. and Yeung, D. (2013). A regularization approach to learning task relationships in multitask learning. *ACM Trans. Knowl. Discov. Data*, 8(3):12:1–12:31.

Zhang, Y., Yeung, D., and Xu, Q. (2010). Probabilistic multi-task feature selection. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A., editors, *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 2559–2567. Curran Associates, Inc.

Zhou, J., Chen, J., and Ye, J. (2011). Clustered multi-task learning via alternating structure optimization. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F. C. N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain*, pages 702–710.

Zweig, A. and Weinshall, D. (2013). Hierarchical regularization cascade for joint learning. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 37–45. JMLR.org.