Learning Kernels for Multiple Predictive Tasks

Submitted in partial fulfillment of the requirements
of the degree of

Doctor of Philosophy

by

Pratik Jawanpuria
(Roll No. 09405012)

Supervisor :
Prof. J. Saketha Nath

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY BOMBAY
2014

To My Grandparents

Thesis Approval

This thesis entitled **Learning Kernels for Multiple Predictive Tasks** by **Pratik Jawanpuria** is approved for the degree of Doctor of Philosophy.

Examiners

____

____

____

Supervisor

____

Chairman

____

Date: ______________

Place: ______________

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

____

(Signature)

____

(Name of the Student)

____

(Roll No.)

Date: ______________

Abstract

In this thesis, we consider the paradigm of training several *related* learning problems (tasks) jointly. The goal is to synergize the related tasks by appropriate sharing of feature information within them and obtain a better generalization across all the tasks (as compared to learning them independently). A key concern while training multiple related tasks jointly is to learn which feature information should be shared among which tasks. Either sharing unhelpful feature information among related tasks or sharing any information with unrelated tasks may result in worsening of performance. The focus of this thesis revolves around this issue. Overall, we present a family of regularized risk minimization based convex formulations, of increasing generality, for learning features (kernels) in various settings involving multiple tasks.

We begin with the question of *which features to share* among all the tasks. This popular setting assumes that all the tasks are related and the aim here is to learn a common feature representation shared by all the tasks. We pose this problem as that of learning a shared kernel. The kernel based framework allows us to implicitly learn task correlations in generic inner product feature spaces. The shared kernel is modeled as a linear combination of a given set of base kernels, the Multiple Kernel Learning setting. We propose a mixed-norm based formulation for learning the shared kernel as well as the prediction functions of all the tasks. We build on this approach and proceed to learn the shared kernel in a couple of richer feature spaces. Firstly, we consider a case where the tasks share low-dimensional subspaces lying in the induced feature spaces of the base kernels. We pose this problem as a mixed Schatten-norm regularized formulation that learns an optimal linear combination of such subspaces. Secondly, we consider learning the shared kernel as a polynomial combination of base kernels. In this setup, the space of possible combinations is exponential in the number of base kernels and we propose a graph based regularizer to perform a clever search in this huge space.

We next study the setting where all the tasks need not share a common feature representation. We develop a formulation that learns groups of related tasks in a *given* feature space. Each group of tasks is represented by a kernel and the problem of learning groups of related tasks is posed as a kernel learning problem. We generalize this setup and propose a formulation that learns which feature spaces are shared among which groups of tasks. We show that the overall performance improves considerably by learning this information.

We end this thesis by discussing our exploration on the problem of computing the kernel selection path in the Multiple Kernel Learning (MKL) framework. Kernel selection path is the variation in the classification accuracy as the fraction of selected kernels is varied from null to unity. We study a generic family of MKL formulations and propose a methodology for computing their kernel selection path efficiently.

In addition to the models developed for learning kernels for multiple tasks, derivations of specialized dual formulations as well as the algorithms proposed for solving them efficiently are also an important contribution of this thesis. In particular, solving mixed Schatten-norm regularized problems as well as searching for optimal polynomial kernel combination in exponentially large search space call for novel optimization methodologies. We show utility of the proposed formulations in a few learning applications such as Rule Ensemble Learning where the goal is to construct an ensemble of conjunctive propositional rules.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this thesis, we study settings that require learning multiple related prediction problems. As a motivating example, consider the problem of recognizing handwritten characters of different writers. Modeling the writing style of each writer may be viewed as a separate problem. One of the simplest approaches for this setting is to pool the data of all the writers together and learn a single prediction function. Since the model parameters learnt is common for all the writers, this approach assumes that the writers have the same writing style. However, this assumption is generally not true. Figure 1.1 shows a large variation in the letter ‘a’ written by 40 different writers [101].

The writing style of people usually depend on factors like right or left handedness, hand-eye co-ordination, motor control, vision, physical features, childhood education or environment, *etc*. Hence, a more preferable solution is to learn a separate prediction function for each writer, using only the writer-specific examples. In this methodology, all the prediction functions are learnt independently. Though it seems to satisfy the concerns raised by the former alternative, this approach rests on the assumption that sufficient amount of training examples are available to learn the prediction functions having good generalization ability. This may not always be the case since personalized data is usually scarce.

Note that both the above methodologies enjoy a distinct advantage. The former method has more training examples to learn from while the latter learns writer-specific prediction functions. Now consider a learning approach that learns writer-specific prediction functions but *shares relevant information* among the problems during the training phase. The intuition be-

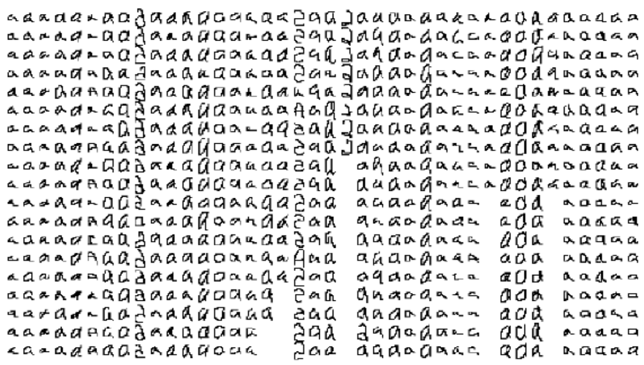
Figure 1.1: The letter a written by 40 different people. This data was collected by Rob Kassel at the MIT Spoken Language Systems Group.

hind information sharing is that the problems are *related* (all of them model handwriting styles) and hence, they may exhibit certain commonalities. For example, all of them will give importance to features like curvature of lines, number of strokes, contiguity of characters, *etc*. Sharing information about such common features may lead to better models since such features are learnt using the training examples of all the writers. In addition to the common features, writer-specific characteristics like right or left handedness, motor control, *etc*., also exist. These may influence the degree of importance given to the features in the writer-specific prediction functions. This paradigm of learning several related tasks *jointly* is popularly known as Multi-task Learning (MTL).

Multi-task Learning [29] aims at sharing the knowledge gained by several related tasks during the training phase. A learning problem is termed as a task. The motive is to improve the generalization achieved across all the tasks as compared to learning them independently. An important precondition for MTL to give better results is that the tasks should be *related*. It usually implies that some helpful correlations should exist among the tasks that allow the flow of a positive knowledge transfer among the tasks during the training stage. For example, the tasks may share a latent feature representation [18, 5, 8], common hyper-parameters [17, 63, 48], *etc*. Sharing information among unrelated tasks may result in the loss of generalization ability and worsening of performance [77, 82]. A detailed discussion on different kinds of task-relatedness studied in the literature is presented in Chapter 2.1.

Multi-task learning setting is quite common in real world applications. We already discussed the handwriting recognition application above. As another example, consider the problem of recognizing different objects. Let each task be that of identifying an object in the images. Note that the images of objects share a common set of features, perhaps those describing lines, shapes, textures, *etc.*, which are different from the input features described in pixels. It is important to discover such shared feature representations that lead to improved recognition ability [61, 122, 5]. Recommender systems for movies, book, *etc.*, seek to predict a user's rating on new/unrated products. The underlying learning algorithm here needs to model the user's preferences. They employ techniques like collaborative filtering [141, 8, 87], which are based on the common observation that users having similar rating history tend to share similar views on new products. Hence, the preferences of several users are modeled jointly. In the application of email spam filtering [132], classifying emails of a user as spam or not-spam is a task. Since every user has a (slightly) different distribution over spam or not-spam emails, each user requires a personalized spam classifier. However, some aspects of relevant or junk emails are usually common across users (or groups of users) and can be learnt jointly. MTL has also been employed in other learning domains like natural language processing [5, 67, 37], computational biology [24, 152], web search ranking [31], *etc.* In addition, learning settings like multi-label classification or multi-class classification may also be viewed under the MTL framework wherein identifying examples associated with a label or class corresponds to a task.

As discussed above, MTL advocates sharing helpful information among related tasks. Hence, a key challenge in MTL is to learn which feature information is to be shared among which tasks. The focus of this thesis revolves around this issue. A large number of existing works [5, 4, 9, 8, 144, 33, 92] assume that all the tasks share a common set of relevant features, in the given (input) feature space. The quality of the final feature representation, in such setups, depends on the quality of the given features. In this thesis, we propose to alleviate this risk of employing low quality features by considering an enriched input space induced by multiple base kernels. A kernel is a function that maps the given feature space to a generic inner product feature space [113]. Hence, the feature learning problem may be equivalently posed as that of learning a suitable kernel function. Much work has been done in the past on learning good quality kernel function for a single prediction task. However, to the best of our knowledge, limited progress has been made on learning kernel functions for multiple tasks. Overall, we present a family of regularized risk minimization based convex formulations, of increasing generality, for learning kernels in various settings involving multiple tasks.

We begin with the question of *which features to share* among the tasks and explore the problem of learning a common feature representation shared across all the tasks. We pose this problem as that of learning a shared kernel function. This shared kernel is learnt as a sparse linear combination of a given set of base kernels, the Multiple Kernel Learning setting. A generic mixed-norm based multi-task learning formulation is proposed that learns the shared kernel as well as the task specific prediction functions. We build on this setup and proceed to learn a common feature representation in a couple of richer feature spaces. Firstly, we consider the setting where the tasks share a latent feature space [29, 18, 8]. We model the latent feature space as a linear combination of low-dimensional orthonormal transformation of the induced feature spaces of the given base kernels. We pose the problem of learning this latent feature space as a mixed Schatten-norm regularized formulation. Secondly, we explore the problem of learning polynomial combination of the given base kernels. Such combinations may be encoded in a directed acyclic graph (DAG). Note that this is a computationally challenging setting since the size of such DAGs can become exponential in the number of base kernels. We model this problem by a graph based regularizer that induces structured sparsity within the DAG. This helps in developing an efficient algorithm to search such DAGs efficiently. The regularizer also enables the proposed formulation to be employed in the application of Rule Ensemble Learning where the goal is to construct an ensemble of conjunctive propositional rules.

The above setups assume that all the tasks share a common feature representation. However, this may not hold in real world applications [70]. Hence, we proceed to a more general setup where it is not assumed that all the tasks are related. In addition, it is also unknown which tasks are related.

We study the problem of learning with whom (tasks) to share the information with, in a given feature space. Note that since feature space is given in this setting, the tasks share information about their model parameters (task parameters) [48]. The information shared within a group of tasks is represented by a kernel function and the problem is posed as a kernel learning problem. We generalize this setup to the case where multiple candidate feature spaces are given and any group of tasks may share a combination of such feature spaces. In particular, we investigate the following setting: some relevant feature spaces are shared across few tasks. We propose a novel multi-task regularizer that leverages commonalities in feature space within sets of tasks, wherever they exist, without paying the penalty when a set of tasks do not share any feature space. Hence, we learn which feature spaces are shared among which tasks and show that the overall performance improves considerably by learning this information.

We end this thesis by discussing our exploration on the problem of tuning p in the p-norm Multiple Kernel Learning (MKL) framework, which is employed in the proposed MTL formulations. The value of p governs the sparsity of the learnt feature space and tuning it for the application at hand is essential for obtaining a good generalization performance. In our experiments, we observed significant variation of performance of MKL based classifiers as p is varied. A simple approach to select the most suitable p could be cross validation. However, state-of-the-art MKL algorithms are too computationally expensive to be invoked hundreds of times to determine the most appropriate p, especially on large scale data sets. We develop a MKL formulation and algorithm that efficiently determine the kernel selection path — variation of classification accuracy as the parameter p is varied from an initial value $p_0 > 1$ to unity.

In addition to the models developed for feature induction in multi-task learning setups, derivations of specialized dual formulations as well as the algorithms thereof for solving the proposed models efficiently are also an important contribution of this thesis. In particular, solving mixed Schatten-norm regularized problems as well as searching for optimal non-linear kernel combination in an exponentially large space call for novel optimization methodologies. The next section details the main contribution of this thesis.

1.1 Main Contributions

In this thesis, we make the following contributions in the field of Multi-task Learning and Multiple Kernel Learning (MKL):

- A MKL based feature induction framework for multiple tasks is proposed. We pose the problem of learning a shared feature representation among the tasks as that of learning a shared kernel. The proposed formulation reduces the excessive dependency on good quality input features by considering an enriched feature space induced by multiple base kernels. Based on an input parameter, the proposed formulation induces varying degree of task-relatedness: a) all the tasks give the same importance to the selected kernels, and b) the tasks share the same set of selected kernels. However, the relative importance of the selected kernels may vary among the tasks.

- A multi-task learning formulation that learns a latent feature space shared by all the tasks. It searches for a sparse linear combination of low-dimensional subspaces residing in the induced feature spaces of the base kernels. The sparse combination constraint helps in discarding the kernels (and corresponding feature spaces) that encourages noisy/low-quality correlations among the tasks. The formulation is posed as a $\ell_1/\ell_{q\geq 1}$ mixed Schatten-norm regularized problem. Such problems are non-standard in literature and call for novel optimization methodologies. Hence, an efficient mirror descent [21, 19, 99] based algorithm is proposed to solve the optimization problem.

- A framework for learning the kernel shared among multiple tasks as a polynomial combination of base kernels is proposed. This space of polynomial combination is exponential in the number of base kernels. We propose a graph based regularizer that induces structured sparsity in this space, which is exploited by an efficient active set algorithm to perform a clever search. We illustrate the efficacy of the proposed formulation in the application of Rule Ensemble Learning, which aims at learning an optimal ensemble of conjunctive rules.

- A multi-task learning formulation that learns kernels (feature spaces) shared by various groups of tasks. It is not known beforehand which tasks share a feature space and hence the proposed formulation explores the space of all possible groups of tasks. We propose a novel multi-task regularizer that encodes the following important structure among the groups of tasks leading to an efficient algorithm for solving it: if a group of tasks do not share any kernel, then none of its superset shares any kernel. We show that the overall performance improves considerably by learning which kernels are shared among which tasks.

- We propose a novel conjecture which states that, for certain p-norm MKL formulations, the number of base kernels selected in the optimal solution monotonically decreases as

p is decreased from an initial value to unity. We prove the conjecture, for a generic family of kernel target alignment based formulations, and show that the kernel weights themselves decay (grow) monotonically once they are below (above) a certain threshold at optimality. This allows us to develop a path following algorithm that systematically generates optimal kernel sets of decreasing size. The proposed algorithm sets certain kernel weights directly to zero for potentially large intervals of p thereby significantly reducing optimization costs while simultaneously providing approximation guarantees. We empirically demonstrate the efficacy of our algorithm, both in terms of generalization ability as well as time taken to compute the entire kernel selection path.

1.2 Outline

The rest of the thesis is organized as follows. In Chapter 2, we provide an overview of Multi-task Learning setting, Multiple Kernel Learning framework and the mirror descent algorithm. Chapter 3 describes our MKL based feature induction framework for multiple tasks. In Chapter 4, we discuss our multi-task formulation that learns a latent feature space shared by all the tasks. In Chapter 5, we present our work on learning non-linear kernel combinations for multiple tasks and its application in Rule Ensemble Learning setting. Chapter 6 discusses our multi-task formulation that learns which kernels are shared among which tasks. We present our analysis of the kernel selection path for p-norm MKL formulation in Chapter 7. We conclude by presenting a summary of this thesis and some conclusions in Chapter 8.

Chapter 2

Background

In this chapter, we present a brief overview of Multi-task Learning (MTL), the Multiple Kernel Learning (MKL) framework and the mirror descent algorithm.

2.1 Multi-task Learning

We begin by considering the standard supervised learning setup in which the goal is to learn a function F that predicts an output $y \in \mathcal{Y}$ for a input $\mathbf{x} \in \mathcal{X}$. We are provided with a set of observations (the training data): $\mathcal{D} = \{(\mathbf{x}_i, y_i), \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}, i = 1, \ldots, m\}$. Here $(\mathbf{x}_i, y_i)$ represents the i^{th} input/output pair. It is desirable that the function F has a good generalization over unseen data as well.

This setup is now extended to multiple tasks. Consider a set of learning tasks, $\mathcal{T}$, such that $|\mathcal{T}| = T$. The training data for the t^{th} task is denoted by: $\mathcal{D}_t = \{(\mathbf{x}_{ti}, y_{ti}), \mathbf{x}_{ti} \in \mathcal{X}_t, y_{ti} \in \mathcal{Y}_t,\ i = 1, \ldots, m_t\}\ \forall t = 1, \ldots, T$, where $(\mathbf{x}_{ti}, y_{ti})$ represents the i^{th} input/output pair of the t^{th} task. The aim is to learn a function F_t, corresponding to each task $t \in \mathcal{T}$, that generalize well over $\mathcal{D}_t$ and its underlying distribution.

One way to learn multiple tasks is to compute each F_t separately and independently. This essentially implies that the t^{th} task only has observations belonging to the set $\mathcal{D}_t$ for guiding the learning process. Note that in setups where the multiple tasks may be *related*, this methodology does not exploit any existing task relatedness. Moreover, in domains where data is scarce for several tasks, learning the decision functions ($F_t\ \forall\ t \in \mathcal{T}$) independently may not yield a good generalization. Another approach could be to aggregate all the tasks and learn just a function common for all the tasks. This is feasible in certain setups where the input spaces ($\mathcal{X}_t$) and the output spaces ($\mathcal{Y}_t$) are identical and all the tasks are homogeneous (for example all are classification tasks or all are regression tasks). However, this methodology makes the setting too simplistic as it may ignore the important differences among the underlying distributions of the tasks.

Multi-task Learning (MTL) [29, 17, 18] paradigm advocates a middle path between the above two extreme approaches for setups, when the given tasks are *related*. It aims to improve the generalization performance by jointly learning a shared representation common across all the tasks. At the same time, tasks specific representations are also learnt. As commented in one of the earliest works on MTL [29] : *Multitask Learning is an approach to inductive transfer that improves generalization by using the domain information contained in the training signals of related tasks as an inductive bias.* Hence, in MTL, though each task learns its own distinct prediction function, the search for the optimal $F_{t'}$ corresponding to task $t'(\in \mathcal{T})$ is guided by the training data of all the tasks ($\mathcal{D}_t \; \forall \; t \in \mathcal{T}$).

In this thesis, we consider affine decision functions for each task. Thus, $F_t(\mathbf{x}) = \langle f_t, \phi(\mathbf{x})\rangle - b_t$, where f_t is the task parameters, $\phi(\cdot)$ is a feature map and b_t is the bias term. In order to learn the task parameters (and the bias terms), we employ the well-established approach of regularized risk minimization [125] and consider the following problem:

$$\min_{f_t, b_t \forall t \in \mathcal{T}} \Omega(f_1, \ldots, f_T)^2 + C \sum_{t=1}^{T} \sum_{i=1}^{m_t} V(F_t(\mathbf{x}_{ti}), y_{ti}) \tag{2.1}$$

where $\Omega(f_1, \ldots, f_T)^2$ is a norm based regularizer, $V(\cdot, \cdot)$ is a suitable convex loss function (for example the hinge loss or the square loss) and $C > 0$ is the regularization parameter. The choice of regularizer ($\Omega(\cdot)$) depends on some prior knowledge of the relationship existing among the tasks. In order to learn the tasks separately and independently[1] in this setup, one may substitute: $\Omega(f_1, \ldots, f_T)^2 = \sum_t \Omega(f_t)^2$. In the following section, we provide a brief description of the task-relatedness notions and the corresponding choice of regularizations employed in existing works. Though the discussion will primarily be focused on the MTL setups based on 2.1, we

[1]The learning of tasks is independent except for the shared parameter C. This can also be overcome by employing task specific C_t instead of C in (2.1).

will also discuss the relevant works in other discriminative or generative settings as well.

2.1.1 Modeling Task Relationship

An important precondition for MTL to improve the generalization performance is to have the given tasks to be *related*. It essentially implies that some helpful correlations exist among the tasks that enable them to *help* each other during the training/learning phase. Various notions of task relatedness have been proposed in the existing works, with empirical success. Few notions of task relatedness have also been shown to have a theoretical foundation as well. In the following section, we provide a brief description of the existing task relatedness notions.

One of the most popular notions of task-relatedness employed in several works [18, 48, 68] is as follows: the task parameters (f_t) of all the tasks are assumed to be *close* to a common (unknown) mean. This notion of task-relatedness originates from the hierarchical Bayesian treatment of MTL [62, 63, 14, 139, 66]. A popular Bayesian approach to enforce this task relatedness is to assume that the task parameters ($f_t \; \forall \; t \in \mathcal{T}$) are generated from a common probability distribution (say Gaussian), having a mean h_0 and covariance matrix Σ. The degree of similarity among the task parameters is determined by the covariance matrix. From the regularized risk minimization perspective, [48] proposed to entail the above task relatedness via a regularizer that penalizes the *distance* of the task parameters from the mean. They proposed to employ the following regularization:

$$\Omega(f_1, \ldots, f_T)^2 = \mu \|h_0\|_2^2 + \sum_{t=1}^{T} \|h_t\|_2^2 \tag{2.2}$$

where $f_t = h_0 + h_t \; \forall \; t = 1, \ldots, T$ and $\mu > 0$ is the parameter that controls the trade-off between regularizing the mean vector h_0 and the variance among the task parameters. It was also shown by [48] that the above task relatedness can be encoded via a kernel. In the rest of the thesis, we refer to this notion of task-relatedness as *Close-by Task Parameters*. Works such as [47, 34] extend this notion to settings where a graph based structural relationship among the tasks is also available as input.

Several works have also focused on learning a common feature representation that is shared across all the tasks [29, 17, 14, 5, 20, 8]. Early works in this setting [17, 62, 63, 18, 14] typically employed a Bayesian approach with neural network machinery to learn a set of 'hidden features' that are common across all the tasks. The hidden layer was learnt using the training sample from all the tasks. They model the prediction function of the t^{th} task as $F_t(\mathbf{x}) = f_t^\top B\mathbf{x}$, where B is the matrix learnt via the hidden layer and is common across all the tasks. Expectation maximization algorithm is usually employed to learn the task parameters.

A similar setting in this regard is that the task parameters share a *simultaneously sparse structure*: only a small number of input features are relevant for every task and the set of such relevant features is common across all the tasks. In regularized risk minimization setting, this is usually accomplished by a multi-task extension of the Group Lasso formulation [142] via the ℓ_1/ℓ_2 block-norm [92, 102] or the ℓ_1/ℓ_∞ block-norm [124, 98]. Thus, with the prediction function being $F_t(\mathbf{x}) = f_t^\top \mathbf{x} - b_t$, the regularizer employed is a ℓ_1/ℓ_q block-norm

$$\Omega(f_1, \ldots, f_T) = \sum_{i=1}^{d} \left(\sum_{t=1}^{T} |f_{ti}|^q \right)^{\frac{1}{q}}, \tag{2.3}$$

where the input feature space is assumed to be d dimensional and f_{ti} denotes the i^{th} parameter variable of the t^{th} task. In addition to sparse feature selection, the ℓ_1/ℓ_∞ block-norm also promotes low variance among the task parameters.

Works such as [5, 144, 8] propose to search for a suitable low dimensional subspace of the given input feature space, common across all the tasks. In particular, [5] considers the setting where $\mathcal{X}_t = \mathbb{R}^d$ and models the prediction function as $F_t(\mathbf{x}) = f_t^\top \mathbf{x} = w_t^\top \mathbf{x} + v_t^\top U\mathbf{x}$. Here U is a (to be learnt) linear low dimensional map common across the tasks and is constrained to be a matrix with orthonormal rows, *i.e.*, $UU^\top = I$. Regularization over the tasks parameters w_t govern the extent of task relatedness among the tasks as lower value of $\sum_t \|w_t\|_2$ imply greater degree of subspace sharing. [144] employ a hierarchical Bayesian approach to model a similar prediction function (as in [5]). In order to learn a low dimensional subspace, they propose to put a super Gaussian distribution like the Laplace distribution over the columns of U. On the other hand, [8] models the prediction function as $F_t(\mathbf{x}) = f_t^\top U\mathbf{x}$, where $U \in \mathbb{R}^{d\times d}$ is a orthonormal matrix that needs to be learnt. Hence, the feature map $\mathbf{x} \to U\mathbf{x}$ may be viewed as a rotation of the original orthonormal basis. In order to learn a low dimensional subspace, [8] employ the ℓ_1/ℓ_2 block-norm regularization (2.3) over the task parameters (f_t). Thus, a sparse feature representation is learnt in the new transformed feature space.

The penalty (2.3) strictly enforces a common sparse structure on all the tasks — either a feature is selected for all the tasks or not selected for any task. This may be a bit too restrictive in real world applications [70, 93, 151]. Recent works aim at developing models that promote a more flexible sparsity structure among the task parameters. [78, 34] address the case when the structure among the tasks can be represented as a tree or a graph. Such prior information may be available in a few applications (such as the genetic association mapping in bio-informatics) from the domain experts. They propose a graph-based group lasso regularizer for such a setting. Though it obtains flexible pattern of sparsity, prior knowledge such as structure among the tasks are usually unknown in most real world applications. A popular framework to obtain flexible sparsity pattern is to employ decomposable task parameters and obtain the final sparsity pattern as the superposition of the individual ones. For instance, [70] proposed to decompose task parameters as $f_t = w_t + v_t \; \forall \; t \in \mathcal{T}$, while [93] proposed $f_t = w \odot v_t \; \forall \; t \in \mathcal{T}$. The symbol $\odot$ represents element-wise product symbol. The parameters v_t are learnt locally (from the task specific examples) and the parameters w/w_t are learnt globally (from the examples of all the tasks). [70] employ the ℓ_1/ℓ_∞ block-norm regularization (2.3) over w_t and ℓ_1-norm over v_t. On the other hand [93] regularize both set of parameters by ℓ_1-norm.

Recent works have attempted to model the co-variance among the task parameters ($f_t \; \forall \; t \in \mathcal{T}$) as well as co-variance among the feature loadings across tasks ($[f_{1i}, \ldots, f_{Ti}] \; \forall \; i$) via matrix-variate normal distribution [145, 147, 7]. In particular, [145] adopts the regularized risk minimization framework, where both the co-variance matrices are learnt as sparse matrices. Learning sparse co-variance matrices help in avoiding overfitting by removing spurious correlations among the tasks and among the features. [7] work upon a similar model in a hierarchical Bayesian setting. In [72], maximum entropy discrimination framework is used to learn a common sparse feature representation over multiple tasks.

Clustered Multi-task Learning

In many applications, tasks may have a specialized group structure where models of tasks within a group are strongly similar to each other and weakly similar to those outside the group. The actual group structure of the tasks is not usually known apriori. For example, in application related to face recognition of different people, we expect that faces of persons belonging to same ethnicity/region/gender share more common characteristics [121]. Similarly, in genomics, organisms from same family (in terms of taxonomy) are expected to show more similar characteristics/behavior [136]. We may also have situations where few of the tasks are 'outliers' (not related to any other task). In such cases, learning all tasks simultaneously may lead to loss of generalization ability and worsening of performance [77, 82]. Many recent works have proposed models for such a setting, known as Clustered Multi-task Learning (CMTL). Works such as [139, 68] propose CMTL formulations that cluster tasks based on the notion of close by task parameters. It implies that the tasks parameters (f_t) of the tasks within a cluster are *closer* to each other than to the tasks outside the cluster. [68] learn clusters of *similar* tasks via a convex relaxation of K-means on the tasks while [139] learn clusters of tasks in a non-parametric hierarchical Bayesian setting where the priors are drawn from the Dirichlet Process. [77] propose to cluster tasks that share a low dimensional feature representation, the task relatedness definition being similar to [8]. They formulate the problem of clustering tasks via mixed-integer programming. Note that the methods [139, 68, 77] perform hard clustering, *i.e.*, each task belong to only one cluster. This may be a stringent constraint in many real world applications.

Recent works have also focused on learning overlapping groups of tasks. [82] model the task parameters as a sparse linear combination of latent basis tasks, *i.e.* $f_t = Ls_t$ where columns of matrix L represent the latent basis tasks. Both L and s_t ($\forall\, t$) are learnt and s_t are regularized by sparsity inducing ℓ_1-norm. The degree of similarity between any two tasks t_1 and t_2 is determined by the extent of overlap between the sparsity patterns of s_{t_1} and s_{t_2}. Tasks having same sparsity pattern are interpreted as belonging to the same group.

2.2 Multiple Kernel Learning

Consider a learning problem like classification or regression and let its training data be denoted by $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1, \ldots, m \mid \mathbf{x}_i \in \mathcal{X}, y_i \in \mathbb{R}\ \forall i\}$, $(\mathbf{x}_i, y_i)$ representing the i^{th} input-output pair. The aim is to learn an affine prediction function $F(\mathbf{x})$ that generalizes well on unseen data. Given a positive definite kernel k that induces a feature map $\phi_k(\cdot)$, the prediction function can be written as: $F(\mathbf{x}) = \langle f, \phi_k(\mathbf{x})\rangle_{\mathcal{H}_k} - b$. Here $\mathcal{H}_k$ is the Reproducing Kernel Hilbert Space (RKHS) [113] associated with the kernel k, endowed with an inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}_k}$, and $f \in \mathcal{H}_k, b \in \mathbb{R}$ are the model parameters to be learnt. A popular framework to learn these model parameters is the regularized risk minimization [125], which considers the following problem

$$\min_{f\in\mathcal{H}_k, b\in\mathbb{R}} \frac{1}{2}\Omega(f)^2 + C\sum_{i=1}^{m}\ell(y_i, F(\mathbf{x}_i)) \tag{2.4}$$

where $\Omega(\cdot)$ is a norm based regularizer, $\ell : \mathbb{R}\times\mathbb{R}\rightarrow\mathbb{R}^+$ is a suitable convex loss function and C is a regularization parameter. From the *representer theorem* [113], we know that the optimal f has the following form $f(\cdot) = \sum_i^m \alpha_i k(\cdot, \mathbf{x}_i)$ where $\alpha = (\alpha_i)_{i=1}^m$ is a vector of coefficients to be learnt. As an example, the support vector machine (SVM) [125] employs $\Omega(f) = \|f\|_{\mathcal{H}_k}$.

As can be observed from above, kernel definition plays a crucial role in defining the quality of the solution obtained by solving (2.4). Hence learning a kernel suitable to the problem at hand has been an active area of research over the past few years. One way of learning kernels is via the Multiple Kernel Learning (MKL) framework [83, 12], in which the kernel k is learnt as a conic combination of the given base kernels $k^1, \ldots, k^n$: $k = \sum_{i=1}^n \eta_i k^i, \eta_i \geq 0, \forall\, i = 1, \ldots, n$. Here $\eta = (\eta_i)_{i=1}^l$ is a coefficient vector to be additionally learnt in the optimization problem (2.4). In this setting, the feature map with respect to the kernel k is given by $\phi_k = (\sqrt{\eta_i}\phi_{k^i})_{i=1}^n$ [see 108, for details]. It is a weighted concatenation of feature maps induced by base kernels. Hence, sparse kernel weights will result in a low dimensional ϕ_k. Some of the additional constraints on η explored by the MKL research community are: ℓ_1-norm constraint [83, 12, 108], ℓ_p-norm constraint ($p > 1$) [81, 128], *etc.* The ℓ_1-norm constraint over kernel weights results in sparse kernel combination and this formulation is usually referred to as ℓ_1-MKL. Works such as [118, 108, 69] have proposed efficient algorithms to solve ℓ_1-MKL formulation, employing cutting planes or gradient descent based methods.

ℓ_p-norm Multiple Kernel Learning

Multiple Kernel Learning formulation with $\ell_{p>1}$-norm constraint over the kernel weights is commonly referred to as ℓ_p-MKL. It has received a lot of attention in the recent literature [39, 128, 104, 81, 103, 69]. As mentioned earlier, in this thesis, we have also employed the ℓ_p-MKL framework to learn sparse combination of kernels (for multiple tasks). In this regard, note that as ℓ_p norm ($p > 1$) rarely induces sparsity since it is differentiable [120]. However, as $p \rightarrow 1$, it promotes only a few leading terms due to the high curvatures of such norms [119]. In order to obtain sparse solutions in such cases, thresholding is a common strategy employed by the existing ℓ_p-MKL ($p > 1$) algorithms such as SMO-MKL [128], OBSCURE [104], UFO-MKL [103] and SPG-GMKL [69]. Hence, in the rest of the thesis, the sparsity inducing behavior of ℓ_p-MKL is stated in this respect. We have also employed thresholding in our experiments.

2.2.1 Hierarchical Kernel Learning

This section introduces the notations and provides a brief description of the HKL setup and formulation [11, 10]. Let the training data be denoted by $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1, \ldots, m \mid \mathbf{x}_i \in \mathcal{X}, y_i \in \mathbb{R}\ \forall i\}$ where $(\mathbf{x}_i, y_i)$ represents the i^{th} input-output pair. As mentioned earlier, in addition to the training data, a DAG and several base kernels embedded on the DAG are provided. Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be the given DAG with $\mathcal{V}$ denoting the set of vertices and $\mathcal{E}$ denoting the set of edges. The DAG structure entails relationships like parent, child, ancestor and descendant [38]. Let $D(v)$ and $A(v)$ represent the set of descendants and ancestors of the node v in the $\mathcal{G}$. It is assumed that both $D(v)$ and $A(v)$ include the node v. For any subset of nodes $\mathcal{W} \subset \mathcal{V}$, the $hull$ and $sources$ of $\mathcal{W}$ are defined as:

$$hull(\mathcal{W}) = \bigcup_{w \in \mathcal{W}} A(w), \quad sources(\mathcal{W}) = \{w \in \mathcal{W} \mid A(w) \cap \mathcal{W} = \{w\}\}.$$

Let $|\mathcal{W}|$ and $\mathcal{W}^c$ denote the size and the complement of $\mathcal{W}$ respectively. Let $k_v : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be the *positive definite* kernel associated with the vertex $v \in \mathcal{V}$. Let $\mathcal{H}_{k_v}$ be its associated RKHS, and ϕ_{k_v} be the feature map induced by it. Given this, HKL employs the following prediction function:

$$F(\mathbf{x}) = \sum_{v \in \mathcal{V}} \langle f_v, \phi_{k_v}(\mathbf{x}) \rangle_{\mathcal{H}_{k_v}} - b,$$

which is an affine model parametrized by $f \equiv (f_v)_{v \in \mathcal{V}}$, the tuple with entries as $f_v, v \in \mathcal{V}$, and $b \in \mathbb{R}$. Some more notations follow: for any subset of nodes $\mathcal{W} \subset \mathcal{V}$, $f_{\mathcal{W}}$ and $\phi_{\mathcal{W}}$ denote the tuples $f_{\mathcal{W}} = (f_v)_{v \in \mathcal{W}}$ and $\phi_{\mathcal{W}} = (\phi_v)_{v \in \mathcal{W}}$ respectively. In general, the entries in a vector are referred to using an appropriate subscript, *i.e.*, entries in $\mathbf{u} \in \mathbb{R}^d$ are denoted by $u_1, \ldots, u_d$ etc. $\mathbf{1}$ refers to an appropriately sized column vector with entries as unity. $\mathbf{Y}$ denotes the diagonal matrix with entries as $[y_1, \ldots, y_m]$. The kernels are denoted by the lower case 'k' and the corresponding kernel matrices are denoted by the upper case 'K'.

HKL formulates the problem of learning the optimal prediction function F as the following regularized risk minimization problem:

$$\min_{f,b} \frac{1}{2}\left(\sum_{v\in\mathcal{V}} d_v \|f_{D(v)}\|_2\right)^2 + C\sum_{i=1}^{m} \ell\left(y_i, F(\mathbf{x}_i)\right) \tag{2.5}$$

where

$$\|f_{D(v)}\|_2 = \left(\sum_{w\in D(v)} \|f_w\|^2\right)^{\frac{1}{2}} \quad \forall\, v\in\mathcal{V}$$

Here, $\ell : \mathbb{R}\times\mathbb{R}\rightarrow\mathbb{R}^+$ is a suitable convex loss function and $(d_v)_{v\in\mathcal{V}}$ are non-negative user-specified parameters.

As is clear from (2.5), HKL employs a ℓ_1/ℓ_2 block-norm regularizer, which is known for promoting group sparsity [142]. Its implications are discussed in the following. For most of $v\in\mathcal{V}$, $\|f_{D(v)}\|_2 = 0$ at optimality due to the sparsity inducing nature of the ℓ_1-norm. Moreover, $(\|f_{D(v)}\|_2 = 0) \Rightarrow (f_w = 0\,\forall\, w\in D(v))$. Thus, it is expected that most of the f_v will be zero at optimality. This implies that the optimal prediction function involves very few kernels. Under mild conditions on the kernels (being strictly positive), it can be shown that this hierarchical penalization induces the following sparsity pattern: $(f_v \neq 0) \Rightarrow (f_w \neq 0\,\forall\, w\in A(v))$. In other words, if the prediction function employs a kernel k_v, then it has to employ all the kernels k_w, where $w\in A(v)$.

[11] propose to solve the following equivalent variational formulation:

$$\min_{\gamma\in\Delta_{|\mathcal{V}|,1}} \min_{f,b} \frac{1}{2}\sum_{w\in\mathcal{V}} \delta_w(\gamma)^{-1}\|f_w\|^2 + C\sum_{i=1}^{m} \ell\left(y_i, F(\mathbf{x}_i)\right), \tag{2.6}$$

where $\mathbf{z}\in\Delta_{s,r} \Leftrightarrow \{\mathbf{z}\in\mathbb{R}^s \mid \mathbf{z}\geq 0, \sum_{i=1}^{s}\mathbf{z}_i^r \leq 1\}$ and $\delta_w(\gamma)^{-1} = \sum_{v\in A(w)} \frac{d_v^2}{\gamma_v}$. From the representer theorem [113], it follows that the effective kernel employed is: $k = \sum_{v\in\mathcal{V}} \delta_v(\gamma) k_v$. Because (2.6) performs an ℓ_1-norm regularization of γ, at optimality most of the $\gamma_v = 0$. Moreover, $\delta_v(\gamma)$, the weight for the kernel k_v, is zero whenever $\gamma_w = 0$ for any $w\in A(v)$. Thus, HKL performs a sparse selection of kernels and can be understood as a specialization of the MKL framework. More importantly, the sparsity pattern for the kernels has the following restriction: if a kernel is not selected then none of the kernels associated with its descendants are selected, as $(\gamma_v = 0) \Rightarrow (\delta_w(\gamma) = 0\,\forall\, w\in D(v))$. For the case of strictly positive kernels, it follows that a kernel is selected only if all the kernels associated with its ancestors are selected. Clearly, this restriction may lead to a large number of kernels being selected.

Since the size of γ is same as that of $\mathcal{V}$ and since the optimal γ is known to be sparse, [11] proposes an active-set algorithm [85] for solving the partial dual in (2.6). At each iteration of the active-set algorithm, the minimization problem in (2.6) is solved with respect to only those variables (γ) in the active set using the projected gradient descent algorithm.

The key advantage of HKL, as presented in [11], is that in performing non-linear feature selection. For example, consider the case where the input space is $\mathcal{X} = \mathbb{R}^n$ and let I be power set of $\{1, \ldots, n\}$. Consider the following 2^n kernels arranged on the usual subset lattice: $k_i(\mathbf{x}, \mathbf{x}') = \Pi_{j \in i} \mathbf{x}_j \mathbf{x}'_j \; \forall \; i \in I$. Now, HKL can be applied in this setup to select the promising sub-products of the input features over all possible sub-products. Please refer to [11] for more such pragmatic examples of kernels and corresponding DAGs. The most interesting result in [11] is that, in all these examples where the size of the DAG is exponentially large, the computational complexity of the active-set algorithm is polynomial in the training set dimensions and the active-set size. Importantly, the complexity is independent of $|\mathcal{V}|$!

2.3 Mirror Descent Algorithm

This section presents the Mirror-Descent (MD) algorithm which is suitable for solving problems of the form $\min_{x \in X} f(x)$ where X is a convex compact set and f is a convex and Lipschitz continuous function on X. It is assumed that an oracle which can compute the sub-gradient (∇f) of f at any point in X is available (an oracle for computing f is not necessary). Interestingly, both the proposed formulations can indeed be posed as convex minimization problems over convex compact sets and oracles for computing the sub-gradient of the objective exist. Hence mirror-descent algorithm can be employed for solving them efficiently.

MD is close in spirit to the projected gradient descent algorithm where the update rule is $x^{(l+1)} = \Pi_X(x^{(l)} - s_l \nabla f(x^{(l)}))$ where Π_X denotes projection onto set X and $x^{(l)}, s_l$ are the iterate value and step-size in the l^{th} iteration respectively. Note that the update rule is equivalent to $x^{(l+1)} = \arg\min_{x \in X} x^\top \nabla f(x^{(l)}) + \frac{1}{2s_l}\|x - x^{(l)}\|_2^2$. The interpretation of this rule is: minimize a local linear approximation of the function while penalizing deviations from current iterate (as the linear approximation is valid only locally). The step-size takes the role of regularization parameter. The key idea in mirror-descent is to choose the regularization term which penalizes deviations from current iterate in such a way that this per-step optimization problem is easy to solve; leading to computationally efficient gradient descent procedures. For convergence guarantees to hold, the regularizer needs to be chosen as a Bregman divergence, *i.e.*, $\frac{1}{2}\|x-x^{(l)}\|_2^2$ is replaced by some Bregman divergence term: $\omega_{x^{(l)}}(x) = \omega(x) - \omega(x^{(l)}) - \nabla\omega(x^{(l)})^\top(x - x^{(l)})$ where ω is the (strongly convex) generating function of the Bregman divergence employed. The step-sizes[2] can be chosen as any divergent series for e.g., $s_l = \frac{1}{l^p}, 0 \le p \le 1$. Note that in case ω is chosen as $\omega(x) = \frac{1}{2}\|x\|_2^2$, then the projected gradient descent algorithm is recovered. The per-step minimization problem mentioned above can now be re-written in terms of ω as:

$$x^{(l+1)} = \text{arg}\min_{x \in X} x^\top \zeta^{(l)} + \omega(x) \tag{2.7}$$

where $\zeta^{(l)} = s_l \nabla f(x^{(l)}) - \nabla\omega(x^{(l)})$. As mentioned above, the strongly convex function ω is chosen cleverly based on X such that (2.7) turns out to be an easy problem to solve. There are two well-known cases where the MD algorithm almost achieves the information-based optimal rates of convergence [99]: i) X is a simplex in $\mathbb{R}^d$ (i.e., $X = \{\mathbf{x} \in \mathbb{R}^d \mid x_i \ge 0,\ \sum_{i=1}^d x_i = 1\}$) and ω is chosen as the negative entropy function: $\omega(\mathbf{x}) = \sum_{i=1}^d x_i \log(x_i)$ ii) X is a spectrahedron (i.e., set of all symmetric positive semi-definite matrices with unit trace) and $\omega(x) = \text{Trace}(x \log(x))$. In fact, in the former case, the per-step problem (2.7) has an analytical solution:

$$x_i^{(l+1)} = \frac{\exp\{-\zeta_i^{(l)}\}}{\sum_{j=1}^d \exp\{-\zeta_j^{(l)}\}} \tag{2.8}$$

Also for this case, the number of iterations can be shown to grow as $log(d)$ and hence nearly-independent of the dimensionality of the problem.

[2]Refer [99] for notes on choosing step-sizes optimally.

Chapter 3

Generic Kernel Learning Framework for Multiple Tasks

Learning a shared feature representation has been a common approach in Multi-task Learning (MTL), as discussed in Chapter 2.1.1. Several works [124, 144, 98, 92, 102] aim at learning a set of input features common across all the tasks by employing ℓ_1/ℓ_2 or ℓ_1/ℓ_∞ block-norm regularization (2.3). In such approaches, the final set of shared features usually belong to the given input space [124, 98, 70, 93]. Thus, the choice of the given input space plays a significant role in them. Moreover, many assume that the degree of influence of the shared feature representation on every task is same. [124, 143, 5, 8]. As an example, if the shared features are given weights then they are *identical* across all the tasks. One way of imparting more flexibility in such a setting is by allowing task-specific weights for the shared features.

In this chapter, we propose a novel framework to learn the shared feature representation across the tasks. We alleviate this risk of employing low quality input features by considering an enriched input space induced by multiple base kernels. Kernels allow us to implicitly learn task relatedness in generic inner product feature spaces. We pose the problem of learning a shared representation among the tasks as that of learning a shared kernel. The latter is learnt as a linear combination of the base kernels, the Multiple Kernel Learning (MKL) setting.

Contributions

We propose a generic ℓ_q/ℓ_p $(1 \leq q \leq 2, p \geq 2)$ mixed norm convex regularizer for learning multiple tasks. Within the feature space induced by each base kernel, the task parameters are regularized by a p-norm. There exists an overall q-norm over the base kernels. As discussed in Section 2.2, for $q \in [1, 2)$, the q-norm promotes sparse selection of the base kernels. Depending on the value of the parameter p, the proposed formulation induces varying degree of task-relatedness:

- Case $p = 2$: The tasks share the feature space induced by the resultant kernel, which is learnt as a sparse linear combination of the base kernels. The tasks are assumed to be equally related.
- Case $p > 2$: The tasks share the small set of relevant base kernels. The feature space learnt for each task is a function of both shared as well as task-specific parameters.

We propose a mirror descent algorithm (Section 2.3) and a sequential minimal optimization (SMO) algorithm to solve the formulation efficiently.

Outline

The rest of the chapter is organized as follows. Section 3.2 formalizes the notation and describes the problem setup. In Section 3.3, we present the generic mixed norm based formulation and discuss it in detail. Efficient algorithms for solving it are presented in Section 3.4. Experimental results are discussed in Section 3.5. We conclude by summarizing the work and the key contributions. We start by discussing the related works in the next section.

3.1 Related Work

In this section, we specifically discuss those approaches that employed the kernel machinery to learn multiple tasks. For a general discussion on the works based on learning a shared feature representation, please refer to Chapter 2.1.1.

[94] proposed a Reproducing Kernel Hilbert Spaces (RKHS) framework for multi-task problems. They introduced matrix-valued kernel functions to learn multiple tasks simultane-

ously. [48, 47] constructed such matrix-valued multi-task kernels that enforced two relationships among the tasks. Firstly, all the tasks shared the given feature space. Secondly, the model parameters of all the tasks were constrained to be centered around a common mean parameter (also discussed in Chapter 2.1.1). In another work, [28] provides various characterization of multi-task kernels that induces universal features [96]. However, the above works did not focus on kernel learning.

Kernel learning for multiple tasks is a relatively less explored setting. [72] proposed to learn a common shared kernel as conic combination of given base kernels in the maximum entropy discrimination based framework. Another work, [8], aim at learning a latent low-dimensional subspace of a given kernel induced feature space that is common across all the tasks. Recently, [109] proposed a ℓ_q/ℓ_p mixed-norm based MTL regularizer with $q \in (0, 1]$. The q-norm promotes sparse kernel combination for $q = 1$ and highly sparse kernel combination for $q \in (0, 1)$. However, in the latter case, their formulation is non-convex. [153] proposed an exclusive-lasso regularizer for multi-task setting with multiple base kernels. They model the setting when each kernel is exclusive to a single task, *i.e.*, the tasks are negatively correlated with respect to the kernels.

3.2 Problem Setup

Consider a set $\mathcal{T}$ of learning tasks, T in number. The training data for the t^{th} task is denoted by: $\mathcal{D}_t = \{(\mathbf{x}_{ti}, y_{ti}),\ i = 1, \ldots, m\}\ \forall t = 1, \ldots, T$, where $(\mathbf{x}_{ti}, y_{ti})$ represents the i^{th} input/output pair of the t^{th} task. For the sake of notational simplicity, we assume that the number of training examples is same across all the tasks. The task predictors are assumed to be affine: $F_t(\mathbf{x}) = \langle f_t, \phi(\mathbf{x})\rangle - b_t,\ t = 1, \ldots, T$, where f_t is the task parameter of the t^{th} task, $\phi(\cdot)$ is the feature map and b_t is the bias term. As stated previously in Chapter 2.1, we follow the regularized risk minimization setup [125] and consider the following problem:

$$\min_{f_t, b_t\ \forall t \in \mathcal{T}} \Omega(f_1, \ldots, f_T)^2 + C \sum_{t=1}^{T} \sum_{i=1}^{m} V(F_t(\mathbf{x}_{ti}), y_{ti}) \tag{3.1}$$

where $\Omega(f_1, \ldots, f_T)^2$ is the regularizer, $V(\cdot, \cdot)$ is a suitable convex loss function (*e.g.* hinge loss, square loss *etc.*) and $C > 0$ is the regularization parameter.

Let $k^1, \dots, k^n$ be the given base kernels and let $\phi^j(\cdot)$ denote the feature map induced by the kernel k^j. Hence, $\phi(\mathbf{x}) = (\phi^1(\mathbf{x}), \dots, \phi^n(\mathbf{x}))$. Let f_t^j represent the projection of f_t onto the ϕ^j space. In other words, $f_t = (f_t^1, \dots, f_t^n)$. With this notation, the prediction function for the task t can be rewritten as $F_t(\mathbf{x}) = \langle f_t, \phi(\mathbf{x})\rangle - b_t = \sum_{j=1}^n \langle f_t^j, \phi^j(\mathbf{x})\rangle - b_t$.

Some more notations before we proceed. Symbols $\mathbf{1}$ and $\mathbf{0}$ refers to an appropriately sized column vector with entries as unity and zero respectively. The kernels are denoted by the lower case 'k' and the corresponding gram matrices are denoted by the upper case 'K'. Domain $\Delta_{s,r}$ ($s \in \mathbb{N}, r > 0$) is defined as: $\mathbf{z} \in \Delta_{s,r} \equiv \{\mathbf{z} \in \mathbb{R}^s \mid \mathbf{z} \geq 0, \sum_{i=1}^s \mathbf{z}_i^r \leq 1\}$. Domain $S(\mathbf{w}, C)$ ($\mathbf{w} \in \mathbb{R}^s, s \in \mathbb{N}, C > 0$) is defined as $\mathbf{z} \in S(\mathbf{w}, C) \equiv \{\mathbf{z} \in \mathbb{R}^s \mid \mathbf{0} \leq \mathbf{z} \leq C\mathbf{1}, \mathbf{z}^\top \mathbf{w} = 0\}$.

3.3 Generic Multi-task Learning Formulation

In this section, we discuss our generic framework on learning kernels for multiple tasks. We propose the following regularizer in (3.1), termed as Multiple Kernel Multi-task Feature Learning (**MK-MTFL**)

$$\Omega_A(f_1, \dots, f_T)^2 = \left(\sum_{j=1}^n \left(\sum_{t=1}^T \|f_t^j\|_2^p \right)^{\frac{q}{p}} \right)^{\frac{2}{q}} \tag{3.2}$$

where $q \in [1, 2]$ and $p \geq 2$. The above regularizer employs mixed-norm over the RKHS norms (i.e. $\|f_t^j\|_2$): a p-norm over the tasks (within each kernel space) and a q-norm over the kernels. In the following, we discuss the implications of the proposed regularization. For $q \in [1, 2)$, the above regularizer promotes kernel selection. This helps in removing noisy feature[1] spaces from the final decision function. The RKHS norms of all the tasks corresponding to the j^{th} kernel, $\|f_t^j\|_2 \; \forall \, t$, are regularized by a p-norm. In particular, for $p = 2$ the tasks share a common learnt kernel while for $p > 2$, the tasks share few relevant base kernels. This will be discussed in more detail during the derivation of the dual formulation. To summarize, the proposed multi-task formulation is:

$$\min_{f_t, b_t \; \forall t \in \mathcal{T}} \frac{1}{2} \Omega_A(f_1, \dots, f_T)^2 + C \sum_{t=1}^T \sum_{i=1}^m V(F_t(\mathbf{x}_{ti}), y_{ti}) \tag{3.3}$$

[1]When $q = 2$, it can be shown that all the kernels are selected with equal weights. Since this is uninteresting, we focus only on $q \in [1, 2)$ in the main text. However, all our derivations will go through for this case as well.

The proposed formulation can also be understood as an adaptation of the composite absolute penalties family studied in [148, 120, 1] to the current problem.

In order to solve (3.3) efficiently, we first rewrite the formulation in a convenient but equivalent form. In the following, we will derive a variational characterization of $\Omega_A(f_1, \ldots, f_T)$ followed by a dual formulation of (3.3) with respect to $f_t, b_t \; \forall \; t$. The dual will provide us new insights into the nature of shared representation among the tasks. To keep the notations simple, the dual is stated for the case where every task is a binary classification problem ($y_{ti} = \{-1, 1\} \; \forall \; i, t$) and the loss function $V(\cdot, \cdot)$ is the hinge loss. However, one can easily extend these ideas to other loss functions and learning settings.

Partial Dual Formulation of (3.3)

We note the following variational characterization of $\Omega_A(f_1, \ldots, f_T)$:

Lemma 3.3.1. *Given $\Omega_A(f_1, \ldots, f_T)$ as defined in (3.2), $q \in [1, 2]$ and $p \geq 2$ we have*

$$\Omega_A(f_1, \ldots, f_T)^2 = \min_{\gamma \in \Delta_{n,\hat{q}}} \max_{\lambda_j \in \Delta_{T,\hat{p}} \forall j} \sum_{j=1}^{n} \frac{\sum_{t=1}^{T} \lambda_{jt} \|f_t^j\|_2^2}{\gamma_j} \tag{3.4}$$

where $\hat{q} = \frac{q}{2-q}$ and $\hat{p} = \frac{p}{p-2}$.

Proof of the above lemma follows from the application of lemma A.1.1 (in Appendix A.1) and the notion of dual norm [26]. Note that in the limiting case of $p = 2, \hat{p} = \infty$ and the optimal value of any λ_{jt} is unity. The following lemma states a dual of (3.3) with respect to $f_t, b_t \; \forall \; t$.

Lemma 3.3.2. *Consider problem (3.3) with the regularizer term replaced with its variational characterization provided in (3.4), $q \in [1, 2)$, $p \geq 2$ and the loss function as the hinge loss $V(F_t(\mathbf{x}_{ti}), y_{ti}) = \max(0, 1 - y_{ti}F_t(\mathbf{x}_{ti}))$. Then the following is a dual of it with respect to the variables $f_t, b_t \; \forall \; t$:*

$$\min_{\gamma \in \Delta_{n,\hat{q}}} \max_{\lambda_j \in \Delta_{T,\hat{p}} \forall j} \max_{\alpha_t \in S(\mathbf{y}_t, C) \forall t} g(\alpha, \lambda, \gamma) \tag{3.5}$$

where

$$g(\alpha, \lambda, \gamma) = \sum_{t=1}^{T} \left[\mathbf{1}^\top \alpha_t - \frac{1}{2} \alpha_t^\top \mathbf{Y}_t \left(\sum_{j=1}^{n} \frac{\gamma_j K_t^j}{\lambda_{jt}} \right) \mathbf{Y}_t \alpha_t \right] \tag{3.6}$$

$\alpha = [\alpha_1^\top, \ldots, \alpha_t^\top]^\top$, $\mathbf{Y}_t$ is the diagonal matrix corresponding to $\mathbf{y}_t$ and $K_t^j \in \mathbb{R}^{m\times m}$ represents the gram matrix of the t^{th} task corresponding to the kernel k^j. The prediction function of the t^{th} task is given by:

$$F_t(\mathbf{x}) = \sum_{i=1}^{m} \bar{\alpha}_{ti} y_{ti} \sum_{j=1}^{n} \frac{\bar{\gamma}_j k^j(\mathbf{x}_{ti}, \mathbf{x})}{\bar{\lambda}_{jt}}$$

where $(\bar{\alpha}_t, \bar{\lambda}, \bar{\gamma})$ is the optimal solution of (3.5).

Proof. The proof follows from the representer theorem [113]. □

The dual in (3.5) reveal further information on the nature of shared representation among the tasks. (3.5) is equivalent to solving regular SVM [125] problems, one for each task, with an effective kernel matrix $\left(\sum_{j=1}^{n} \frac{\gamma_j K_t^j}{\lambda_{jt}}\right)$. This is well defined since it can be shown that $\bar{\lambda}_{jt} > 0 \; \forall \; j, t$ if gram matrices K_{tj} are positive[2] definite. The variables γ are shared across the tasks. In particular, $\bar{\gamma}_j = 0$ imply that the kernel k^j is absent from the shared feature representation and vice-versa. As $q \to 1$ ($\hat{q} \to 1$), the formulation promotes sparser $\bar{\gamma}$ and hence, sparser combination of the base kernels.

Now consider the case $p = 2$ in (3.5). It can be easily seen from Lemma 3.3.1 that $\bar{\lambda}_{jt} = 1 \; \forall \; j, t$. For each task, the effective kernel becomes $\left(\sum_{j=1}^{n} \gamma_j k^j\right)$. Hence, when $p = 2$, (3.5) induces same feature map across all the tasks. If $p > 2$, the tasks share the same set of active kernels. Let $\mathcal{A} = \{j | \bar{\gamma}_j > 0\}$ be the set of active base kernels at optimality. Then the resultant kernel for the t^{th} task is given by: $\sum_{j\in\mathcal{A}} \frac{\gamma_j k_t^j}{\lambda_{jt}}$. Though the active set ($\mathcal{A}$) is common across the tasks, the feature map induced by the resultant kernels will vary. The active set is governed globally by $\bar{\gamma}$ and $\bar{\lambda}$ have a local effect by influencing the importance given to the selected kernels. Hence, the task relatedness notion captured by the proposed formulation (3.3) is as follows: the tasks share few base kernels. In particular, when $p = 2$, the learnt feature maps across the tasks are *identical*.

In the following section, we propose efficient algorithms to solve (3.5) based on the mirror descent algorithm (2.3) and sequential minimal optimization (SMO) algorithm [105, 128].

[2]A small ridge maybe added in case they are positive semi-definite.

3.4 Optimization Algorithm

In this section, we propose efficient algorithms to solve the dual (3.5). The dual optimization involves three sets of variables: α, λ and γ. The nature of the feasibility sets of γ change with the value of q. Hence, we propose different optimization techniques for different ranges of q.

Case 1: $\mathbf{q = 1}$

In this case, $\hat{q} = 1$ and $\hat{p} \in (1, \infty)$ in (3.5). Instead of solving the min-max problem in (3.5), we prefer to solve equivalently: $\min_{\gamma \in \Delta_{n,\hat{q}}} h(\gamma)$ where

$$h(\gamma) = \max_{\lambda_j \in \Delta_{T,\hat{p}} \forall j} \max_{\alpha_t \in S(\mathbf{y}_t, C) \forall t} g(\alpha, \lambda, \gamma)$$

It is easy to see that $h(\gamma)$ is convex and using Danskin's theorem [23] one can obtain its gradient (∇h) provided $h(\gamma)$ can be computed efficiently for a given γ. Since the feasibility set for this problem is a simplex, the mirror descent algorithm (outlined in Section 2.3) can be employed to solve it very efficiently. Computing $h(\gamma)$ is equivalent to maximization of $g(\lambda, \alpha; \gamma)$ with respect to λ and α for the given γ. This maximization can be performed efficiently using an alternate maximization over the λ and α variables. For a fixed value of λ, maximization over α is equivalent to solving T regular SVM problems. For a fixed value of α the problem of maximization with respect to λ is equivalent to: $-\sum_{j=1}^{k} \min_{\lambda_j \in \Delta_{T,\bar{p}}} \sum_{t=1}^{T} \frac{D_{jt}}{\lambda_{jt}}$ where $D_{jt} = \frac{1}{2}\gamma_j \alpha_t^T \mathbf{Y}_t \mathbf{K}_{tj} \mathbf{Y}_t \alpha_t$. From lemma A.1.1, we know that this problem has an analytical solution.

The per-step computational complexity is dominated by the SVM computations and calculation of effective kernel: $O(nm^2T)$. The number of iterations for the mirror descent grows as $\log(n)$ and in our simulations we found that the alternate minimization typically converges in around 3-5 iterations. Hence, the overall computational complexity is $O(n \log(n) m^2 T)$.

Case 2: $\mathbf{q} \in (\mathbf{1}, \mathbf{2})$

In this case, $\hat{q} \in (1, \infty)$ and $\hat{p} \in (1, \infty)$ in (3.5). The formulation is similar to Case 1 except for the feasibility set over γ: it is now $\hat{q}$-norm ball instead of the relatively simpler 1-norm ball (simplex). One way to proceed is the algorithm described in Case 1 with the mirror descent method replaced by projected gradient descent method [108]. Hence, the overall algorithm will be an outer iteration of projected gradient descent steps with an inner step of alternate maximization algorithm. However, projections onto $\hat{q}$-norm ball are significantly more challenging than those onto simplex [90]. Hence, we propose a significantly efficient alternative in the following. We start by presenting an equivalent dual of the primal (3.3)

$$\max_{\alpha_t \in S(\mathbf{y}_t, C) \forall t} \max_{\lambda_j \in \Delta_{T,\hat{p}} \forall j} \sum_{t=1}^{T} \mathbf{1}^\top \alpha_t - \frac{1}{2} \left(\sum_{j=1}^{n} \left(\sum_{t=1}^{T} \frac{\alpha_t^\top \mathbf{Y}_t K_t^j \mathbf{Y}_t \alpha_t}{\lambda_{jt}} \right)^{\bar{q}} \right)^{\frac{1}{\bar{q}}}$$

where $\bar{q} = \frac{q}{2(q-1)}$. The above equivalent is obtained after interchanging min-max in (3.5) using min-max theorems [110] and subsequent application of the notion of dual norm [26] to eliminate γ. We propose an alternative maximization procedure to solve for the above problem. Given the value of λ, the optimization problem is similar to ℓ_ρ-MKL optimization problem [81]. It can be solved by employing the cutting plane or the SMO algorithm. For a fixed value of α, the maximization over λ has an analytical solution (see the lemma in A.1.1).

3.5 Experiments

In this section, we evaluate the proposed multi-task formulation (3.3) on object recognition datasets. The aim is to compare the generalization ability of the proposed multi-task formulation against the single task learning baselines. We present detailed comparison of the proposed method with state-of-the-art multi-task feature learning (**MTSFL**) algorithm [8] as well as another proposed multi-task learning formulation in Section 4.5. In the current experiments, the singular value decomposition based **MTSFL** algorithm [8] failed to execute due to the large size of the object recognition datasets.

MK-MTFL The proposed multiple kernel multi-task feature learning formulation. The parameter q was set to 1 for sparse kernel selection and three different values of p (the norm over tasks) were considered: $2, 6, 8.67$.

MKL The baseline formulation where each task is learnt using MKL [108] formulation.

SVM The baseline formulation where each task is learnt using an SVM [125].

The following datasets were employed in our comparisons:

Dataset	**MKL**	**SVM**	**MK-MTFL**		
			$p = 2$	$p = 6$	$p = 8.67$
Caltech	52.75	54.35	65.12	65.15	**65.27***
Oxford	83.73	81.67	85.29	**85.98***	85.88

Table 3.1: Comparing the accuracy achieved by various approaches on object categorization datasets. **MK-MTFL** achieves significantly higher accuracies as compared to the **MKL** and **SVM** methods.

Caltech A benchmark multi-class object categorization dataset [49]. A collection of images from 102 categories of objects like faces, watches, ants, *etc.*, with 30 images per category. Each 1-vs-rest binary classification problem was considered as a task (no. tasks=102). The training-test splits and 25 base kernels are available at `http://mkl.ucsd.edu/dataset/ucsd-mit-caltech-101-mkl-dataset`.

Oxford A benchmark multi-class object categorization dataset [100]. Collection of images from 17 varieties of flowers. The number of images per category is 80. Each 1-vs-rest binary classification problem was considered as a task (no. tasks=17). Three predefined training-test splits consisting of 40 training, 20 validation and 20 test examples and seven base kernels are provided with the dataset. The dataset is available at `http://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html`.

Hyper-parameter selection and Base kernels: Parameter C was tuned by three-fold cross validation for all the methods. **SVM** also had a three-fold nested cross-validation for the base kernels. The values of C are considered from the set $\{5e-1, 5, \ldots, 5e+5\}$. We employed the benchmark base kernels for both the datasets.

Evaluation criterion: We report the mean percentage accuracy achieved over 5 standard splits for each dataset. The highest percentage accuracy achieved in each dataset is highlighted. In addition, if the improvement in accuracy is statistically significant (greater than 94% confidence with paired t-test), the result is marked with a * symbol.

The results on object categorization are summarized in Table 3.1. The proposed **MK-MTFL** formulation outperforms the **MKL** and **SVM** methods confirming that it is indeed able to learn a suitable shared feature representation for multiple tasks. Figure 3.1 compares the

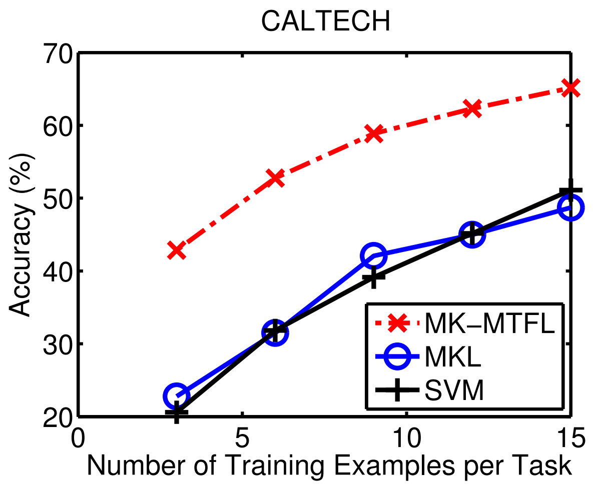

Figure 3.1: Variation in accuracy (%) obtained by multi-task and single task learning methods on the test set as the number of training examples per task is varied in **Caltech** dataset. The advantage of the proposed multi-task approach **MK-MTFL** over **MKL** and **SVM** is more pronounced at low training data region.

learning rates of the three algorithms as a function of training examples in the **Caltech** dataset. We can observe that the advantage of the proposed **MK-MTFL** over the single task learning baselines is quite high (20%) at low training data region. This advantage starts decreasing as the number of training examples increase. This is expected because with more training data, a task should generally be able to obtain good generalization ability on its own. We postpone a more detailed evaluation of **MK-MTFL** to Chapter 4.5, where it is compared with state-of-the-art multi-task feature learning algorithm **MTSFL** [8] as well as another proposed algorithm.

3.6 Conclusions

We propose a generic ℓ_q/ℓ_p mixed-norm based convex regularizer for multi-task learning. The regularizer learns a shared feature representations by learning a common kernel learnt as a sparse linear combination of base kernels. We also propose mirror descent and sequential minimal optimization based algorithms for solving the formulation efficiently. The empirical results show that the proposed methodology is scalable as well as achieves good generalization on benchmark object categorization datasets.

Chapter 4

Latent Feature Representation for Multiple Tasks

In this chapter, we consider the setting where multiple tasks share a latent feature representation. Existing works [5, 4, 144, 8, 33] aim at learning such a latent subspace within a given feature space. In such approaches, the quality of the final feature representation depends on the choice of the given feature space. One way to minimize the risk of employing low quality input features is to work with multiple base kernels (feature spaces). Since some approaches [9, 8] admit a kernel as the given feature space, a simple workaround for them is to employ a kernel that is the average of multiple base kernels. This may be is fine provided all the base kernels are good. In real world setting, unreliable or low quality kernels may also exist among the base kernels. Hence, a more principled solution is to learn the most suitable kernels along with the shared latent subspace within them.

In this chapter, we propose a model to learn the shared latent subspace from multiple base kernels. The proposed formulation searches for suitable low-dimensional orthonormal transformation of the induced feature spaces of the given base kernels. The shared feature space is learnt as a sparse linear combination of such subspaces.

Contributions

We model the multi-task learning problem as a ℓ_1/ℓ_q $(1 \leq q \leq 2)$ mixed Schatten-norm regularized formulation. The formulation learns an optimal combination of low dimensional subspaces from few base kernels. Mixed Schatten-norm regularized problems are non-standard in literature and call for novel optimization methodologies. Hence, we propose an efficient mirror descent based algorithm for solving the proposed formulation, with the feasibility set being a ℓ_1/ℓ_q mixed Schatten-norm ball. Such problems are non-standard in literature and call for novel optimization methodologies. We show that the entropy function can be employed as the regularizer in the context of mirror descent method, leading to an efficient algorithm for solving the proposed formulation.

Outline

The rest of the chapter is organized as follows. Section 4.2 describes the problem setup. The proposed formulation and the mirror descent based algorithm for solving it are presented in Sections 4.3 and 4.4 respectively. Experimental results are discussed in Section 4.5. We conclude by summarizing the work and the key contributions. We begin by presenting an overview of the existing works related to sparse feature induction for multiple tasks.

4.1 Related Work

Several works [5, 4, 144, 8, 33] have proposed algorithms to learn a latent subspace shared across all the tasks (also discussed in Chapter 2.1.1). This problem may be formalized as follows: learn a suitable feature map $\phi(\cdot) : \mathbf{x} \rightarrow U\mathbf{x}$, which is shared across all the tasks. A desirable property of this latent subspace is low-dimensionality (sparse). Works such as [5, 84] fix the dimension of the latent subspace by a user define parameter. On the other hand, [8] propose an orthogonal transformation (*i.e.*, U is constrained to be an orthonormal matrix) and learn sparse latent features by imposing the ℓ_1/ℓ_2 regularization (2.3) over the task parameters. This ensures the simultaneously sparse structure among the task parameters in the latent feature space. [144, 82] model the task parameters as sparse linear combination of a set of basic classifiers. Note that the above methods work on a given input feature space. We propose a kernel learning based multi-task formulation that learns an optimal combination of low dimensional subspaces from multiple kernels. When the number of base kernels is unity, the proposed formulation reduces to the one proposed in [8].

4.2 Problem Setup

We follow the notations introduced in Chapter 3. Consider a set $\mathcal{T}$ of learning tasks, T in number. The training data for the t^{th} task is denoted by: $\mathcal{D}_t = \{(\mathbf{x}_{ti}, y_{ti}),\ i = 1, \ldots, m_t\}\ \forall\, t = 1, \ldots, T$, where $(\mathbf{x}_{ti}, y_{ti})$ represents the i^{th} input/output pair of the t^{th} task. Let $k^1, \ldots, k^n$ be the given base kernels. Let $\phi^j(\cdot)$ denote the feature map induced by the j^{th} kernel k^j. As in case of the multi-task sparse feature learning algorithm [8], **MTSFL**, we construct latent features as orthonormal transforms of the given features, *i.e.*, $\mathbf{L}_j\phi^j(\mathbf{x})$ where $\mathbf{L}_j$ is an (unknown) orthonormal matrix. We represent the set of all orthonormal matrices of dimensionality d as $\mathbf{O}^d$. Let f_t^j represent the projection of the f_t (the task parameter of t^{th} task) onto the ϕ^j space. In other words, $f_t = (f_t^1, \ldots, f_t^n)$. In addition, let the entries of f_t^j be denoted by $f_t^{jl}, l = 1, \ldots, d_j$, where d_j is the dimensionality of the ϕ^j space. The vector with entries $f_t^{jl}, t = 1, \ldots, T$, is denoted by f^{jl}. With the above notations, the prediction function for the task t can be written as $F_t(\mathbf{x}) = \sum_{j=1}^n \langle f_t^j, \mathbf{L}_j\phi^j(\mathbf{x})\rangle - b_t$.

Some more notations before we proceed. Symbols $\mathbf{1}$ and $\mathbf{0}$ refers to an appropriately sized column vector with entries as unity and zero respectively. The kernels are denoted by the lower case 'k' and the corresponding gram matrices are denoted by the upper case 'K'. Domain $\Delta_{s,r}$ $(s \in \mathbb{N}, r > 0)$ is defined as: $\mathbf{z} \in \Delta_{s,r} \equiv \{\mathbf{z} \in \mathbb{R}^s \mid \mathbf{z} \geq 0, \sum_{i=1}^s \mathbf{z}_i^r \leq 1\}$ and for simplicity let $\Delta_{s,1}$ be denoted as Δ_s. Domain $S(\mathbf{w}, C)$ $(\mathbf{w} \in \mathbb{R}^s, s \in \mathbb{N}, C > 0)$ is defined as $\mathbf{z} \in S(\mathbf{w}, C) \equiv \{\mathbf{z} \in \mathbb{R}^s \mid \mathbf{0} \leq \mathbf{z} \leq C\mathbf{1}, \mathbf{z}^\top\mathbf{w} = 0\}$.

4.3 Learning Latent Subspaces in Multiple Kernels

In this section, we present the proposed multiple kernel multi-task sparse feature learning formulation (**MK-MTSFL**). Working within the regularized risk minimization [125] setting (3.1), we propose the following mixed-norm regularizer to learn an optimal combination of low dimensional subspaces from few base kernels across multiple tasks:

$$\Omega_S(f_1, \ldots, f_T)^2 = \left(\sum_{j=1}^{n}\left(\sum_{l=1}^{d_j}\|f^{jl}\|_2\right)^q\right)^{\frac{2}{q}} \tag{4.1}$$

where $q \in [1, 2]$. The above regularizer induces sparsity at two different levels of feature induction: i) for $q \in [1, 2)$, the q-norm promotes sparse combination of base kernels, and ii) the 1-norm over the feature loadings within each kernel induced feature space promotes highly sparse feature selection within each (selected) kernel. The innermost 2-norm over the tasks ensures that the any selected feature is common across all the tasks in the learnt feature space. To summarize, the proposed **MK-MTSFL** formulation is as follows:

$$\min_{f_t, b_t \forall t \in \mathcal{T}} \min_{\mathbf{L}_j \in \mathbf{O}^{d_j} \forall j} \frac{1}{2} \Omega_S(f_1, \ldots, f_T)^2 + C \sum_{t=1}^{T} \sum_{i=1}^{m} V(F_t(\mathbf{x}_{ti}), y_{ti}) \tag{4.2}$$

where $V(\cdot, \cdot)$ is a suitable convex loss function and $C > 0$ is the regularization parameter.

Though the proposed formulation looks similar to that in **MK-MTFL** (3.2) proposed in the previous chapter, there are a few fundamental difference. **MK-MTSFL** (4.1) employs a 1-norm over feature loadings across tasks rather than an RKHS norm over feature loadings for each task present in (3.2). Hence, the learnt feature space corresponding to a selected kernel will be a low dimensional subspace in **MK-MTSFL**, whereas **MK-MTFL** employs the whole feature space of a selected kernel. In addition, **MK-MTSFL** also learn suitable orthonormal transformation of the kernel induced feature spaces.

The proposed regularizer, in its present form, explicitly enumerates the features induced by base kernels, which is infeasible in case of infinite dimensional feature spaces (for example in Gaussian kernels). Hence, in the following, we first derive a variational characterization of $\Omega_S(f_1, \ldots, f_T)$, followed by a kernelized partial dual of (4.2). To keep the notations simple, the partial dual formulation is stated for the case when each task is a binary classification problem ($y_{ti} = \{-1, 1\} \; \forall \, i, t$) and the loss function $V(\cdot, \cdot)$ employed in (4.2) is the hinge loss. However, one can easily extend these ideas to other loss functions and learning problems. The following lemma provides the variational characterization of $\Omega_S(f_1, \ldots, f_T)$:

Lemma 4.3.1. *Given $\Omega_A(f_1, \ldots, f_T)$ as defined in (4.1) with $q \in [1, 2]$, we have*

$$\Omega_S(f_1, \ldots, f_T)^2 = \min_{\lambda \in \Delta_{n, \bar{q}}} \min_{\gamma_j \in \Delta_{d_j} \forall j} \sum_{t=1}^{T} \sum_{j=1}^{n} \sum_{l=1}^{d_j} \frac{(f_t^{jl})^2}{\gamma_{jl} \lambda_j} \tag{4.3}$$

where $\bar{q} = \frac{q}{2-q}$.

Proof. The above can be derived by repeated application of lemma A.1.1. □

Now we perform a change of variables: $\frac{f_t^{jl}}{\sqrt{\gamma_{jl}\lambda_j}} = \bar{f}_t^{jl}$. Also, we define $\bar{f}_t^j$ as vector with entries as $\bar{f}_t^{jl}, l = 1, \ldots, d_j$. Employing these new variables, one can re-write the **MK-MTSFL** formulation (4.2) with the hinge loss as:

$$\min_{\lambda, \gamma_j, \mathbf{L}_j \forall j} \sum_{t=1}^{T} \min_{\bar{f}_t, b_t, \xi_t} \frac{1}{2} \sum_{j=1}^{k} \langle \bar{f}_{tj}, \bar{f}_{tj} \rangle + C \sum_{i=1}^{m_t} \xi_{ti}$$
$$\text{s.t. } y_{ti}\left(\sum_{j=1}^{k} \langle \Lambda_j^{\frac{1}{2}} \bar{f}_{tj}, \mathbf{L}_j \phi^j(\mathbf{x}_{ti}) \rangle - b_t\right) \geq 1 - \xi_{ti}, \ \xi_{ti} \geq 0$$
$$\lambda \in \Delta_{n,\bar{q}}, \gamma_j \in \Delta_{d_j} \ \forall \ j, \mathbf{L}_j \in \mathbf{O}^{d_j} \ \forall \ j$$

where Λ_j is a diagonal matrix with entries[1] as $\lambda_j \gamma_{jl}, l = 1, \ldots, d_j$. The following is the Lagrange dual of the above primal with respect to the variables $\bar{f}_t, b_t, \xi_t \ \forall \ t$:

$$\min_{\lambda, \gamma_j, \mathbf{L}_j \forall j} \sum_{t=1}^{T} \max_{\alpha_t \in S_{m_t}(C) \forall t} \mathbf{1}^\top \alpha_t - \frac{1}{2} \alpha_t^\top \mathbf{Y}_t \left(\sum_{j=1}^{k} \Phi_{tj}^\top \mathbf{L}_j^\top \Lambda_j \mathbf{L}_j \Phi_{tj} \right) \mathbf{Y}_t \alpha_t$$
$$\text{s.t. } \lambda \in \Delta_{n,\bar{q}}, \gamma_j \in \Delta_{d_j} \ \forall \ j, \mathbf{L}_j \in \mathbf{O}^{d_j} \ \forall \ j$$

where Φ_{tj} is the data matrix with columns as $\phi^j(\mathbf{x}_{ti}), i = 1, \ldots, m_t$. Denoting $\mathbf{L}_j^\top \Lambda_j \mathbf{L}_j$ by $\bar{\mathbf{Q}}_j$ and eliminating variables λ, γ and $\mathbf{L}$ leads to the following optimization problem:

$$\min_{\bar{\mathbf{Q}}} \sum_{t=1}^{T} \max_{\alpha_t \in S_{m_t}(C) \forall t} \mathbf{1}^\top \alpha_t - \frac{1}{2} \alpha_t^\top \mathbf{Y}_t \left(\sum_{j=1}^{k} \Phi_{tj}^\top \bar{\mathbf{Q}}_j \Phi_{tj} \right) \mathbf{Y}_t \alpha_t$$
$$\text{s.t. } \bar{\mathbf{Q}}_j \succeq 0, \sum_{j=1}^{k} (trace(\bar{\mathbf{Q}}_j))^{\bar{q}} \leq 1$$

The difficulty in working with this formulation is that the explicit mappings $\phi^j(\cdot) \ \forall \ j$ are required. We now describe a way of overcoming this problem by kernelizing the formulation (also see [8]). Let $\Phi_j \equiv [\Phi_{1j}, \ldots, \Phi_{Tj}]$ and the compact singular value decomposition[2] (SVD) of Φ_j be $\mathbf{U}_j \Sigma_j \mathbf{V}_j^\top$. Now, we introduce new variables $\mathbf{Q}_j$ such that $\bar{\mathbf{Q}}_j = \mathbf{U}_j \mathbf{Q}_j \mathbf{U}_j^\top$. Here, $\mathbf{Q}_j$ is a symmetric positive semi-definite matrix of size same as rank of Φ_j. Eliminating variables

[1]Note that, $\lambda_j \gamma_{jl}$ being zero does not cause a problem since then both f_t^{jl} and $\bar{f}_t^{jl}$ will be zero for all $t = 1, \ldots, T$ at optimality

[2]Since we perform a compact SVD, Σ_j is a square diagonal matrix of size equal to rank (which is upper bounded by $\sum_{t=1}^{T} m_t$) of Φ_j and the number columns of $\mathbf{U}_j, \mathbf{V}j$ are the again equal to the rank.

$\bar{\mathbf{Q}}_j$, we can re-write the above problem using $\mathbf{Q}_j$ as:

$$\min_{\mathbf{Q}} \sum_{t=1}^{T} \max_{\alpha_t \in S_{m_t}(C) \forall t} \mathbf{1}^\top \alpha_t - \frac{1}{2} \alpha_t^\top \mathbf{Y}_t \left(\sum_{j=1}^{k} \mathbf{M}_{tj}^\top \mathbf{Q}_j \mathbf{M}_{tj} \right) \mathbf{Y}_t \alpha_t \tag{4.4}$$

$$\text{s.t. } \mathbf{Q}_j \succeq 0, \sum_{j=1}^{k} (trace(\mathbf{Q}_j))^{\bar{q}} \leq 1$$

where $\mathbf{M}_{tj} = \Sigma_j^{-1} \mathbf{V}_j^\top \Phi_j^\top \Phi_{tj}$. Note that calculation of $\mathbf{M}_{tj}$ does not require the kernel induced features explicitly and hence, the formulation is kernelized.

The partial dual (4.4) provides more insights into the original formulation. Given the $\mathbf{Q}_j$ matrices, the problem is equivalent to solving T regular SVM problems individually. The variables $\mathbf{Q}_j$ are learnt using training examples of all the tasks and are shared across the tasks. The trace constraints are a generalization of the trace-norm regularization and hence promote low rank $\mathbf{Q}_j$ matrices at optimality. Thus, the formulation indeed constructs a shared sparse feature representation using multiple base kernels simultaneously. In the next section, we describe an efficient mirror descent based algorithm for solving this dual formulation.

4.4 Optimization Algorithm

Instead of solving the min-max problem in the dual (4.4), we propose to equivalently solve the following problem:

$$\min_{\mathbf{Q} \in R(\bar{q})} g(\mathbf{Q}) \tag{4.5}$$

where $R(\bar{q}) \equiv \left\{ \mathbf{Q} | \mathbf{Q} = diag(\mathbf{Q}_1, \ldots, \mathbf{Q}_n), \mathbf{Q}_j \succeq 0 \, \forall \, j = 1, \ldots, n, \sum_{j=1}^{n} (trace(\mathbf{Q}_j))^{\bar{q}} = 1 \right\}$, *i.e.*, $\mathbf{Q}$ is a block diagonal matrix, and $g(\mathbf{Q})$ is a function equal to the optimal value of the following optimization problem

$$\max_{\alpha_t \in S_{m_t}(C) \forall t} \sum_{t=1}^{T} \mathbf{1}^\top \alpha_t - \frac{1}{2} trace(\mathbf{QB}) \tag{4.6}$$

where $\mathbf{B}$ is a block diagonal matrix with entries as $\mathbf{B}_j = \sum_{t=1}^{T} \mathbf{M}_{tj} \mathbf{Y}_t \alpha_t \alpha_t^\top \mathbf{Y}_t \mathbf{M}_{tj}^\top$.

Note that this minimization problem (4.5) is an instance of a mixed Schatten-norm based regularized problem. Also, since $g(\cdot)$ is point-wise maximum over affine functions in $\mathbf{Q}$, it is convex in $\mathbf{Q}$. Moreover, the gradient ∇g can be obtained using the Danskin's theorem: $\nabla g(\mathbf{Q}^{(l)}) = -\frac{1}{2}\mathbf{B}^{(l)}$ where $\mathbf{B}^{(l)}$ is the value obtained using optimal α_t obtained while evaluating $g(\mathbf{Q}^{(l)})$. The evaluation of $g(\mathbf{Q})$ is equivalent to solving T regular SVM problems. Hence, the mirror descent algorithm (Chapter 2.3) can be employed for solving (4.5).

As discussed before, mirror descent algorithm employs a projection operator based on Bregman-like distance function. For the mixed Schatten-norm regularized problem at hand, we show that the the strongly convex $\omega(x) = trace(x \log(x))$ is a suitable Bregman generating function. With this choice, the per-step optimization problem (2.7) becomes:

$$\min_{\mathbf{Q}\in R(\bar{q})} trace(\zeta^{(l)}\mathbf{Q}) + trace(\mathbf{Q}\log(\mathbf{Q})) \tag{4.7}$$

where $\zeta^{(l)} = s_l \nabla g(\mathbf{Q}^{(l)}) - \nabla\omega(\mathbf{Q}^{(l)})$. We already noted how Danskin's theorem can be employed to obtain $\nabla g(\mathbf{Q}^{(l)})$. Also, $\nabla\omega(\mathbf{Q}^{(l)}) = \log(\mathbf{Q}^{(l)}) + \mathbf{I}$ where $\mathbf{I}$ is the identity matrix of appropriate size. Note that both $\mathbf{Q}$ and $\zeta^{(l)}$ share the same block diagonal structure. In the following we argue that at optimality, $\mathbf{Q}$ and $\zeta^{(l)}$ share the same eigen-vectors (also refer [8] for the case of spectrahedron geometry). Let the eigen value decomposition (EVD) of the matrix $\zeta^{(l)}$ be $\mathbf{Z}\Pi\mathbf{Z}^\top$ (here Π is the diagonal matrix containing eigen values). Passing from variable $\mathbf{Q}$ to Θ according to $\mathbf{Q} = \mathbf{Z}\Theta\mathbf{Z}^\top$, we can re-write the optimization problem (4.7) as: $\min_{\Theta\in R(\bar{q})} trace(\Pi\Theta) + trace(\Theta\log(\Theta))$. It is easy to see that the unique optimal solution of this (strongly convex) problem is a diagonal matrix: for every diagonal matrix $\mathbf{D}$ with entries ± 1 and every feasible solution Θ, the quantity $\mathbf{D}\Theta\mathbf{D}$ will remain feasible and moreover achieves the same objective (since Π is diagonal). It follows that the optimal set of solutions must be invariant with respect to $\Theta \to \mathbf{D}\Theta\mathbf{D}$ transformations, which is possible if and only if Θ is also diagonal (else uniqueness of the solution breaks-down). If the entries in Π, Θ are $\pi_{jl}, \theta_{jl}, l = 1, \ldots, r_j, j = 1, \ldots, n$ (here r_j represents the dimension of matrix $\mathbf{Q}_j$) respectively, then the above problem is equivalent to:

$$\min_{\theta} \sum_{j=1}^{n}\sum_{l=1}^{r_j} \left(\theta_{jl}\log(\theta_{jl}) + \theta_{jl}\pi_{jl}\right) \tag{4.8}$$
$$\text{s.t. } \theta_{jl} \geq 0 \forall j, \sum_{j=1}^{n}(\sum_{l=1}^{r_j}\theta_{jl})^{\bar{q}} \leq 1$$

Though this is a convex problem and involves vectorial variables, the number of variables can be as large as $n\sum_{t=1}^{T} m_t$. Hence, it is not wise to employ standard optimization toolboxes to solve this problem. In fact one can reduce the number of variables to n by performing the following trick: introduce variables $\rho_j, j = 1, \dots, n$ and re-write problem (4.8) as:

$$\min_{\rho} \sum_{j=1}^{n} \min_{\theta_j} \sum_{l=1}^{r_j} \left(\theta_{jl} \log(\theta_{jl}) + \theta_{jl}\pi_{jl}\right) \tag{4.9}$$
$$\text{s.t. } \theta_{jl} \geq 0, \rho_j \geq 0 \; \forall \, j, l, \sum_{l=1}^{r_j} \theta_{jl} = \rho_j, \sum_{j=1}^{n} \rho_j^{\bar{q}} \leq 1$$

The above minimization problem with respect to θ_j has an analytical solution: $\theta_{jl} = \rho_j \bar{\pi}_{jl}$, where $\bar{\pi}_{jl} = \frac{\exp\{-\pi_{jl}\}}{\sum_{\bar{l}=1}^{r_j} \exp\{-\pi_{j\bar{l}}\}}$. Thus, θ_j can be eliminated from (4.9):

$$\min_{\rho} \sum_{j=1}^{n} \left(\rho_j \log(\rho_j) + \rho_j \left(\sum_{l=1}^{r_j} \left(\pi_{jl}\bar{\pi}_{jl} + \bar{\pi}_{jl} \log(\pi_{jl})\right) \right) \right)$$
$$\text{s.t. } \rho_j \geq 0, \sum_{j=1}^{n} \rho_j^{\bar{q}} \leq 1$$

The size of this problem is n and can be easily managed by standard solvers like `cvx` [56, 55]. Apart from the computational cost of `cvx` (which is negligible for n in order of few hundred), the dominant computations are i) Eigen value decomposition (EVD) of $\zeta^{(l)}$ which involves EVD[3] of n matrices of size $r_j \times r_j$, and ii) solving T regular SVM. This is still reasonable as, in the special case number of base kernels is unity, `cvx` need not be employed and the dominant computation remains the same as that in the alternating minimization algorithm in [8]. In the case $q = 1$, `cvx` need not be employed as the final problem has an analytic solution.

4.5 Experiments

This section summarizes results of simulations that illustrate the merits of the **MK-MTSFL** algorithm as well as the multi-task learning algorithm proposed in Chapter 3 (**MK-MTFL**). We compare their generalization performance against **MTSFL** algorithm [8] and other baselines. We also evaluate the efficiency of the proposed mirror descent algorithm against the alternate minimization algorithm proposed in [8]. We begin with the results comparing the generalization performance of the following formulations:

[3]The computation of $\log(\mathbf{Q}_j)$ can be done efficiently (avoiding EVD) by book-keeping the EVD of $\mathbf{Q}_j$

MK-MTSFL : The multiple kernel multi-task sparse feature learning formulation[4] presented in Section (4.3). Three different values of q-norm were considered: 1.01, 1.5, 1.99.

MK-MTFL : The multiple kernel multi-task feature learning formulation presented in Chapter (3.3). The parameter setting is same as described in Chapter 3.5.

MTSFL : State-of-the-art multi-task sparse feature learning formulation [8]. The original code provided by the authors and is available at `http://ttic.uchicago.edu/~argyriou/code/mtl_feat/mtl_feat.tar`.

MTSFL$_{\mathbf{avg}}$: A simple extension of the **MTSFL** formulation to the case of multiple base kernels. It is same as the **MTSFL** formulation but with input kernel as the average of all the given base kernels.

MKL The baseline formulation where each task is learnt using MKL [108] formulation.

SVM The baseline formulation where each task is learnt using an SVM [125].

The following datasets were considered in our experiments:

School A benchmark multi-task regression dataset [8]. The goal is to predict performance of students given their descriptions/past record. Data for 15362 students from 139 schools is available and each student is described using 28 features. The regression problem of predicting performances in each school is considered as a task. At random 15 examples in each task were taken as training data and the rest as test data. It is available at `http://ttic.uchicago.edu/~argyriou/code/mtl_feat/school_splits.tar`.

Sarcos A multiple-output regression dataset [146]. The objective is to predict inverse dynamics corresponding to the seven degrees-of-freedom of a SARCOS anthropomorphic robot arm based on 21 input features. The dataset comprises of 48933 data points. Prediction of inverse dynamics for each degree-of-freedom is considered as a task. 2000 random examples were sampled in case of each task and 15 of them were used as training examples while the rest were kept aside as test examples. It is available at `http://www.gaussianprocess.org/gpml/data/`.

[4]The code for the proposed formulations, **MK-MTSFL** and **MK-MTFL**, is available at: `http://www.cse.iitb.ac.in/saketh/research/MTFL.tgz`

Parkinsons A multi-task regression dataset from the UCI repository [50]. The objective is to predict two Parkinson's disease symptom scores (motor UPDRS, total UPDRS) for patients based on 19 bio-medical features. The original dataset comprises of 5,875 recordings for 42 patients. The regression problem of predicting each symptom score for each patient is considered as a task. Thus, the total number of tasks is $42 \times 2 = 84$. We took 15 random examples per task for training and the rest were used as test data.

Hyper-parameter selection and Base kernels: In case of **SVM, MTSFL** a three fold nested cross-validation procedure is employed to tune the C and the kernel parameter. For the proposed methods (**MK-MTSFL** and **MK-MTFL**) and **MTSFL**$_{\text{avg}}$, a 3-fold cross-validation procedure is employed to tune C. The values of the C parameter considered are in the set $\{5e-4, 5e-3, \ldots, 5e+2\}$. We employed the one linear and six Gaussian kernels. The six different values of the Gaussian parameter considered were: $\{1e-2, 1e-1, \ldots, 1e+3\}$.

Evaluation criterion: Following [8], percentage explained variance [14] is employed as the measure of performance for regression problems. The explained variance per task can be computed as 1 minus the ratio of mean squared error and the variance of the true outputs on the test dataset. Note that a simple regressor which predicts every output as the mean of the outputs in the dataset achieves an explained variance of zero on the dataset. Thus, if explained variance is negative a simple mean estimate is better than the corresponding regressor and hence the method is of no use. In general, higher explained variance indicate better method. In our case we have multiple tasks and we compute the overall explained variance as the mean of explained variances over each task. Hence we can no longer claim that a methodology which achieves an overall negative score is useless. But still, the higher the explained variance, the better the method. We report the mean percentage explained variance achieved over 10 random splits for each dataset. The highest explained variance achieved in each dataset is highlighted. In addition, if the improvement in explained variance is statistically significant ($\geq 94\%$ confidence with paired t-test), then the result is marked with a * symbol.

Table 4.1 summarizes the results of the experiments when the base kernels were constructed by employing all the features (whole kernels). It clearly shows that the proposed **MK-MTSFL** formulation outperforms every baseline confirming that learning a shared feature representation is indeed beneficial. Moreover the improvement over state-of-the-art multi-task

Datasets	MTSFL	MTSFL$_{\text{avg}}$	MKL	SVM	MK-MTFL			MK-MTSFL		
					$p = 2$	$p = 6$	$p = 8.67$	$q = 1.01$	$q = 1.5$	$q = 1.99$
School	13.88	12.35	-15.42	-8.97	-9.18	-6.60	-4.53	**14.02***	13.86	13.95
Sarcos	-23.28	-40.13	-86.27	-26.16	-82.58	-71.70	-69.94	**26.15***	25.66	25.52
Park.	57.81	42.72	14.16	44.35	-249.32	-455.36	-2156.34	45.37	58.11	**59.27***

Table 4.1: Comparison of percentage explained variance with various methods on multi-task regression datasets with whole kernels (kernels constructed by employing all the features). The proposed **MK-MTSFL** achieves significantly better generalization as compared to the **MTSFL** and other algorithms.

Datasets	MTSFL	MTSFL$_{\text{avg}}$	MKL	SVM	MK-MTFL			MK-MTSFL
					$p = 2$	$p = 6$	$p = 8.67$	$q = 1.5$
School	-	9.76	-3.07	-4999.08	10.95	13.68	**14.35***	-
Sarcos	2.44	37.80	48.72	-14.64	**49.82**	39.23	30.12	38.82
Park.	-	35.98	**60.66***	53.59	-19.58	-15.88	-418.86	-

Table 4.2: Comparison of percentage explained variance with various methods on multi-task regression datasets with feature-wise kernels. The proposed **MK-MTFL** performs significantly better with feature-wise base kernels than with the whole kernels. '-' denote results where the number of base kernels was too large for the multi-task algorithm to generate results.

methodology (**MTSFL**) is statistically significant. This illustrates the advantage of employing multiple kernels in the context of multi-task applications. Note that **MTSFL**$_{avg}$, which also employs multiple kernels by averaging them, achieves less generalization than **MTSFL**. This highlights the benefit of learning sparse combinations of suitable kernels. The results also underline the importance of solving the **MK-MTSFL** formulation at various values of q-norm, as there seems to be no evident optimal value across datasets. Automatic tuning of parameter q calls for further investigation (we discuss it for single task MKL setup in Chapter 7).

From the results in Table 4.1, the **MK-MTFL** formulation seems to be highly incompetent. Note that **MK-MTSFL** learns sparse feature representation within each selected kernel whereas **MK-MTFL** promotes sparsity only at the level of kernel combination. However, **MK-MTFL** can also learn sparse feature representations provided suitable base kernels are employed. For instance, if the base kernels are derived from each individual input features, **MK-MTFL** may indeed learn sparse feature representations. In order to see if the poor performance of **MK-MTFL** is due to inherent limitations in **MK-MTFL** or due to restricted choice of base kernels, we repeated the simulations with the base kernels including those derived from individual input features. The results are reported in Table 4.2. Due to the large increase in the number of

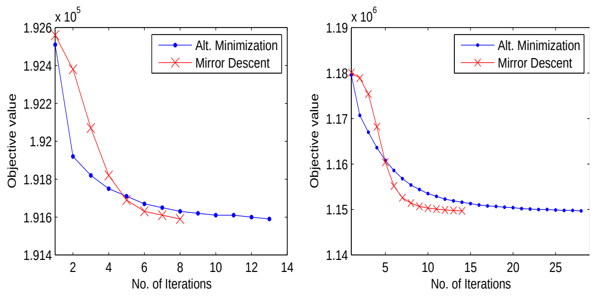

Figure 4.1: Comparison of convergence of the proposed mirror descent methodology and the alternate minimization methodology [8] with $T = 20$ (left) and $T = 139$ (right) respectively on the School dataset. The mirror descent technique achieves faster convergence, especially when the number of tasks is large.

base kernels some formulations failed to execute with the available resources. This failure is indicated with a '-' symbol. Interestingly, the performance of **MK-MTFL** improved by a great magnitude. It is the best performing method in School and Sarcos datasets. In case of the Parkinsons dataset, though its explained variance seems to be low, we observed that the mean squared error is the least among all the methods and the improvement in terms of the mean squared error is statistically significant.

To summarize, in case large number of base kernels are available, then **MK-MTFL** is the suitable choice both from computational and generalization perspective. When a restricted set of base kernels are available, then **MK-MTSFL** is the better option.

The results illustrating the efficacy of the proposed mirror descent methodology are summarized in Figure 4.1. We solved the **MTSFL** formulation [8] using the proposed mirror descent algorithm as well as the alternate minimization [8]. The plots compare their convergence rate on School dataset with $T = 20$ and $T = 139$. It can be seen that the mirror descent algorithm achieves faster convergence (with similar per step computational complexity), especially with large number of tasks.

4.6 Conclusions

We propose multi-task formulation that learns a shared latent feature space within multiple base kernels. A mirror descent based algorithm is developed for solving the proposed mixed Schatten-norm regularized formulation. In the process, we showed that the negative entropy function can be employed as an efficient Bregman generating function for solving such problems via the mirror descent algorithm. Empirical results show that the proposed multi-task formulation achieves better generalization than state-of-the-art and outperforms other baselines as well.

Chapter 5

Learning Multiple Tasks with Hierarchical Kernels

In the previous chapters, we proposed formulations to learn a shared kernel across multiple tasks. In particular, the shared kernel was learnt as a linear combination of either the base kernels (Chapter 3) or latent subspaces within the base kernels (Chapter 4). The base kernels employed for constructing the shared kernel, in both these settings, were assumed to be *unrelated*. In this chapter, we consider a more generic setting where the base kernels may have certain relationship or dependencies among them. We explore a setup where the base kernels have a hierarchical relationship among themselves, *i.e.,* the base kernels can be embedded on a directed acyclic graph. Such a setting occurs naturally in several domains like bio-informatics [114], dictionary learning [73], text processing [91], *etc*. Learning the shared kernel across tasks as a polynomial combination of a given set of kernels is a special case of this setting.

A Multiple Kernel Learning (MKL) [83, 12] framework for construction of sparse linear combinations of base kernels embedded on a directed acyclic graph (DAG) was recently proposed by [11]. Since the DAG induces hierarchical relations between the base kernels, this framework is more commonly known as Hierarchical Kernel Learning[1] (HKL). It has been established that HKL provides a powerful algorithm for task specific non-linear feature selection. HKL employs a carefully designed ℓ_1/ℓ_2 block-norm regularizer: ℓ_1-norm across some predefined components associated with the DAG and ℓ_2-norm within each such component.

[1]HKL framework and formulation has been briefly discussed in Chapter 2.2.1

However, the sparsity pattern of kernel (feature) selection induced by this regularizer is somewhat restricted: *a kernel is selected only if the kernels associated with all its ancestors in the DAG are selected.* In addition, it can be proved that the weight of the kernel associated with a (selected) node will always be greater than the weight of the kernels associated with its descendants. Such a restricted selection pattern and weight bias may limit the applicability of HKL in real world problems.

Contributions

We propose a two-fold generalization of HKL. The first is employing a $\ell_1/\ell_\rho, \rho \in (1,2)$, block-norm regularizer that mitigates the above discussed weight and selection bias among the kernels, henceforth termed as gHKL. Note that for the special case of $\rho = 2$, gHKL renders the HKL regularizer. Further, gHKL is generalized to the paradigm of Multi-task Learning (MTL), where multiple related tasks need to be learnt jointly. We consider the MTL setup where the tasks share a common feature representation [92, 102]. The proposed generalization, henceforth referred to as gHKL$_{\text{MT}}$, learns a shared kernel across the tasks as a sparse combination of the base kernels that are arranged on a DAG. In addition, gHKL$_{\text{MT}}$ is generic enough to model additional correlations existing among the given tasks.

Though employing a $\ell_1/\ell_\rho, \rho \in (1,2)$, regularizer is an incremental modification to HKL formulation, devising an algorithm for solving it is not straight-forward. The projected gradient descent employed in the active set algorithm for solving HKL [11] can no longer be employed for solving gHKL as projections onto ℓ_ρ-norm balls are known to be significantly more challenging than those onto ℓ_1-norm balls [90]. Hence naive extensions of the existing HKL algorithm will not scale well. Further, the computational challenge is compounded with the generalization for learning multiple tasks jointly. The key technical contribution of this work is the derivation of a highly specialized partial dual of gHKL/gHKL$_{\text{MT}}$ formulations and an efficient mirror descent [99] based active set algorithm for solving it. The dual presented here is an elegant convex optimization problem with a Lipschitz continuous objective and constrained over a simplex. Moreover, the gradient of the objective can be obtained by solving a known and well-studied variant of the MKL formulation. This motivates employing the mirror descent algorithm that is known to solve such problems efficiently. Further efficiency is brought in by employing an active set method similar in spirit to that in [11].

In addition, we also focus on the application of Rule Ensemble Learning (REL) [46, 45], where HKL has not been previously explored. Given a set of basic propositional features describing the data, the goal in REL is to construct a compact ensemble of conjunctions with the given propositional features that generalizes well for the problem at hand. Such ensembles are expected to achieve a good trade-off between interpretability and generalization ability. We illustrate the efficacy of gHKL/gHKL$_{\text{MT}}$ formulations in learning a compact set of promising rules.

Outline

The rest of the chapter is organized as follows. Section 5.1 introduces the problem setup. In Section 5.2, we present the proposed gHKL and gHKL$_{\text{MT}}$ formulations. The key technical derivation of the specialized dual is also presented in this section. The proposed mirror descent based active set algorithm for solving gHKL/gHKL$_{\text{MT}}$ formulations is discussed in Section 5.3. In Section 5.4, we propose to solve the REL problem by employing gHKL/gHKL$_{\text{MT}}$ formulations and discuss its details. Experimental results in the REL setting are discussed in Section 5.5. We conclude the chapter by summarizing the work and the key contributions.

5.1 Problem Setup

The kernel setup for the setup for gHKL is the same as that for HKL, described in Section 2.2.1. In summary, a DAG and several base kernels embedded on the DAG are provided. Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be the given DAG with $\mathcal{V}$ denoting the set of vertices and $\mathcal{E}$ denoting the set of edges. Let $D(v)$ and $A(v)$ represent the set of descendants and ancestors of the node v in the $\mathcal{G}$. It is assumed that both $D(v)$ and $A(v)$ include the node v. For any subset of nodes $\mathcal{W} \subset \mathcal{V}$, the $hull$ and $sources$ of $\mathcal{W}$ are defined as:

$$hull(\mathcal{W}) = \bigcup_{w \in \mathcal{W}} A(w), \quad sources(\mathcal{W}) = \{w \in \mathcal{W} \mid A(w) \cap \mathcal{W} = \{w\}\}.$$

Let $|\mathcal{W}|$ and $\mathcal{W}^c$ denote the size and the complement of $\mathcal{W}$ respectively.

Let $k_v : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be the *positive definite* kernel associated with the vertex $v \in \mathcal{V}$. Let $\mathcal{H}_{k_v}$ be its associated RKHS, and ϕ_{k_v} be the feature map induced by it. Given this, gHKL employs the following prediction function:

$$F(\mathbf{x}) = \sum_{v \in \mathcal{V}} \langle f_v, \phi_{k_v}(\mathbf{x}) \rangle_{\mathcal{H}_{k_v}} - b,$$

which is an affine model parametrized by $f \equiv (f_v)_{\ v \in \mathcal{V}}$, the tuple with entries as $f_v, v \in \mathcal{V}$, and $b \in \mathbb{R}$. Some more notations follow: for any subset of nodes $\mathcal{W} \subset \mathcal{V}$, $f_{\mathcal{W}}$ and $\phi_{\mathcal{W}}$ denote the tuples $f_{\mathcal{W}} = (f_v)_{v \in \mathcal{W}}$ and $\phi_{\mathcal{W}} = (\phi_v)_{v \in \mathcal{W}}$ respectively. The kernels are denoted by the lower case 'k' or 'h' and the corresponding kernel matrices are denoted by the upper case 'K' or 'H'.

5.2 Generalized Hierarchical Kernel Learning

In this section, we present the proposed generalizations over HKL — the gHKL and gHKL$_{\text{MT}}$ formulations. We begin by introducing gHKL formulation.

5.2.1 The gHKL Primal Formulation

Recall that the HKL formulation employs a ℓ_1/ℓ_2 block norm regularizer. As we shall understand in more detail later, a key reason for the kernel weight bias problem and the restricted sparsity pattern in the HKL is the ℓ_2-norm regularization. One way to alleviate such restrictions is by employing the following generic regularizer:

$$\Omega_S(f) = \sum_{v \in \mathcal{V}} d_v \|f_{D(v)}\|_\rho \tag{5.1}$$

where $f = (f_v)_{v \in \mathcal{V}}$, $\|f_{D(v)}\|_\rho = \left(\sum_{w \in D(v)} \|f_w\|^\rho\right)^{\frac{1}{\rho}}$ and $\rho \in (1, 2]$. The implications of the ℓ_1/ℓ_ρ block-norm regularization are discussed in the following. Since the ℓ_1-norm promotes sparsity, it follows that $\|f_{D(v)}\|_\rho = 0$ ($f_w = 0 \; \forall \; w \in D(v)$) for most $v \in \mathcal{V}$. This phenomenon is similar as in the HKL. But now, even in cases where $\|f_{D(v)}\|_\rho$ is not forced to zero by the ℓ_1-norm, many components of $f_{D(v)}$ tend to zero[2] (*i.e.*, $f_w \to \mathbf{0}$ for many $w \in D(v)$) as the

[2]Note that as ℓ_ρ ($\rho > 1$) norm is differentiable, it rarely induce true sparsity [120]. However, as $\rho \to 1$, they promote only a few leading terms due to the high curvatures of such norms [119]. In order to obtain a

value of ρ tends to unity. Also note that $\rho = 2$ render the HKL regularizer. To summarize, the proposed gHKL formulation is:

$$\min_{f_v \in \mathcal{H}_{k_v} \forall v \in \mathcal{V}, b \in \mathbb{R}} \frac{1}{2}(\Omega_S(f))^2 + C\sum_{i=1}^{m} \ell\left(y_i, F(\mathbf{x}_i)\right) \tag{5.2}$$

We next present the gHKL$_{\text{MT}}$ formulation, which further generalize gHKL to MTL paradigm.

5.2.2 The gHKL$_{\text{MT}}$ Primal Formulation

We begin by introducing some notations applicable in multi-task learning setup. Let T be the number of tasks and let the training data for the t^{th} task be denoted by $\mathcal{D}_t = \{(\mathbf{x}_{ti}, y_{ti}),\ i = 1, \ldots, m \mid \mathbf{x}_{ti} \in \mathcal{X}, y_{ti} \in \mathbb{R}\ \forall i\}$, where $(\mathbf{x}_{ti}, y_{ti})$ represents the i^{th} input/output pair of the t^{th} task. For the sake of notational simplicity, it is assumed that the number of training examples is the same for all the tasks. The prediction function for the t^{th} task is of the form: $F_t(\mathbf{x}) = \sum_{v \in \mathcal{V}} \langle f_{tv}, \phi_{k_v}(\mathbf{x}) \rangle_{\mathcal{H}_{k_v}} - b_t$, where $f_t = (f_{tv})_{v \in \mathcal{V}}$ and b_t are the task parameters to be learnt for all the tasks. We propose the following regularized risk minimization problem for estimating these task parameters and term it as gHKL$_{\text{MT}}$:

$$\min_{f_t, b_t\ \forall\ t} \frac{1}{2}\left(\underbrace{\sum_{v \in \mathcal{V}} d_v \left(\sum_{w \in D(v)} (Q_w(f_1, \ldots, f_T))^{\rho}\right)^{\frac{1}{\rho}}}_{\Omega_T(f_1, \ldots, f_T)}\right)^2 + C\sum_{t=1}^{T}\sum_{i=1}^{m} \ell(y_{ti}, F_t(\mathbf{x}_{ti})), \tag{5.3}$$

where $\rho \in (1, 2]$ and $Q_w(f_1, \ldots, f_T)$ is a norm-based multi-task regularizer on the task parameters $f_{tw}\ \forall\ t$. In the following, we discuss the effect of the above regularization. Firstly, there is a ℓ_1-norm regularization over each group of nodes (feature spaces) and a ℓ_ρ-norm regularization within each group. This ℓ_1/ℓ_ρ block-norm regularization is same as that of the gHKL and will have the same effect on the sparsity pattern of the selected feature spaces (kernels). Hence, only a few nodes (feature spaces) will be selected by the gHKL$_{\text{MT}}$ regularizer $\Omega_T(f_1, \ldots, f_T)$. Secondly, nature of the task relatedness within each (selected) feature space is governed by the $Q_w(f_1, \ldots, f_T)$ regularizer.

completely sparse solution in such cases, thresholding is a common strategy employed by previous ℓ_p-MKL ($\rho > 1$) algorithms [128, 104, 103, 69]. We employed thresholding in our experiments.

For instance, consider the following definition of $Q_w(f_1, \ldots, f_T)$ [92, 71]

$$Q_w(f_1, \ldots, f_T) = \left(\sum_{t=1}^{T} \|f_{tw}\|^2 \right)^{\frac{1}{2}} \tag{5.4}$$

The above regularizer couples the task parameters within each feature space by a ℓ_2-norm. It encourages the task parameters within a feature space to be either zero or non-zero simultaneously. Therefore, $\Omega_T(f_1, \ldots, f_T)$ based on (5.4) has the following effect: i) all the tasks will simultaneously select or reject a given feature space, and ii) overall only a few feature spaces will be selected in the gHKL-style sparsity pattern.

Several multi-task regularizations [48, 47, 68] have been proposed to promote proximity among task parameters within a given feature space. This correlation among the tasks may be enforced while learning a shared sparse feature space by employing the following $Q_w(f_1, \ldots, f_T)$:

$$Q_w(f_1, \ldots, f_T) = \left(\mu \left\| \frac{1}{T+\mu} \sum_{t=1}^{T} f_{tw} \right\|^2 + \sum_{t=1}^{T} \left\| f_{tw} - \frac{1}{T+\mu} \sum_{t=1}^{T} f_{tw} \right\|^2 \right)^{\frac{1}{2}} \tag{5.5}$$

where $\mu > 0$ is a given parameter. The above $Q_w(f_1, \ldots, f_T)$ consists of two terms: the first regularizes the mean while the second regularizes the variance of the task parameters in the feature space induced by kernel k_w. The parameter μ controls the degree of proximity among the task parameters, with lower μ encouraging higher proximity. Note that when $\mu = \infty$, (5.5) simplifies to (5.4).

The gHKL$_{\text{MT}}$ regularizer $\Omega_T(f_1, \ldots, f_T)$ based on (5.5) has the following effect: i) all the tasks will simultaneously select or reject a given feature space, ii) overall only a few feature spaces will be selected in the gHKL-style sparsity pattern, and iii) within each selected feature space, the task parameters f_{wt} $\forall$ t are in proximity. Thus, the gHKL$_{\text{MT}}$ framework provides mechanism to learn sparse shared features, while preserving task correlation in the learnt feature space. As we shall discuss in the next section, more generic correlations among task parameters may be also modeled using the gHKL$_{\text{MT}}$ formulation.

It is clear that the gHKL optimization problem (5.2) may be viewed as a special case of the gHKL$_{\text{MT}}$ based on (5.4) optimization problem and with the number of tasks set to unity. Hence, the rest of the discussion regarding dual derivation and optimization focuses primarily on the gHKL$_{\text{MT}}$ formulation.

5.2.3 The gHKL$_{\text{MT}}$ Dual Formulation

As mentioned earlier, due to the presence of the ℓ_ρ-norm term in the gHKL$_{\text{MT}}$ formulation, naive extensions of the projected gradient based active set method in [11] will be rendered computationally infeasible on real world datasets. Hence, we first re-write the gHKL$_{\text{MT}}$ formulation in an elegant form, which can then be solved efficiently. To this end, we note the following variational characterization of $\Omega_T(f_1, \ldots, f_T)$:

Lemma 5.2.1. *Given $\Omega_T(f_1, \ldots, f_T)$ and $Q_w(f_1, \ldots, f_T)$ as defined in (5.3) and (5.5) respectively, we have*

$$\Omega_T(f_1, \ldots, f_T)^2 = \min_{\gamma \in \Delta_{|\mathcal{V}|,1}} \min_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in V} \sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda)^{-1} Q_w(f_1, \ldots, f_T)^2 \tag{5.6}$$

where $\delta_w(\gamma, \lambda)^{-1} = \sum_{v \in A(w)} \frac{d_v^2}{\gamma_v \lambda_{vw}}$, $\mathbf{z} \in \Delta_{s,r} \equiv \{\mathbf{z} \in \mathbb{R}^s \mid \mathbf{z} \geq 0, \sum_{i=1}^s \mathbf{z}_i^r \leq 1\}$ and $\hat{\rho} = \frac{\rho}{2-\rho}$.

Note that $\rho \in (1, 2)$ implies $\hat{\rho} \in (1, \infty)$. Proof of the above lemma is provided in appendix A.1.2.

In order to keep the notations simple, in the remainder of this section, it is assumed that the learning tasks at hand are binary classification, *i.e.*, $y_{ti} \in \{-1, 1\} \; \forall \, t, i$ and the loss function is the hinge loss. However, one can easily extend these ideas to other loss functions and learning problems. Refer appendix A.1.8 for the gHKL$_{\text{MT}}$ dual formulation with general convex loss functions.

Lemma 5.2.2. *Consider problem (5.3) with the regularizer term replaced with its variational characterization given in (5.6) and the loss function as the hinge loss $\ell(y, F_t(\mathbf{x})) = \max(0, 1 - yF_t(\mathbf{x}))$. Then the following is a partial dual of it w.r.t. the variables $f_t, b_t \; \forall \, t$:*

$$\min_{\gamma \in \Delta_{|\mathcal{V}|,1}} \min_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} \max_{\alpha_t \in S(\mathbf{y}_t, C) \forall t} G(\gamma, \lambda, \alpha) \tag{5.7}$$

where

$$G(\gamma, \lambda, \alpha) = \mathbf{1}^\top \alpha - \frac{1}{2} \alpha^\top \mathbf{Y} \left(\sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) H_w \right) \mathbf{Y} \alpha$$

$\alpha = [\alpha_1^\top, \ldots, \alpha_T^\top]^\top$, $S(\mathbf{y}_t, C) = \{\beta \in \mathbb{R}^m \mid 0 \leq \beta \leq C, \; \sum_{i=1}^m y_{ti}\beta_i = 0\}$, $\mathbf{y}_t = [y_{t1}, \ldots, y_{tm}]^\top \; \forall \, t$, $\mathbf{Y}$ is the diagonal matrix corresponding to the vector $[\mathbf{y}_1^\top, \ldots, \mathbf{y}_T^\top]^\top$, $\mathbf{1}$ is a $mT \times 1$ vector with entries as unity, $\delta_w(\gamma, \lambda)^{-1} = \sum_{v \in A(w)} \frac{d_v^2}{\gamma_v \lambda_{vw}}$, $\Delta_{n,1} = \{\eta \in \mathbb{R}^n \mid \eta \geq 0, \sum_{i=1}^n \eta_i \leq 1\}$,

$\hat{\rho} = \frac{\rho}{2-\rho}$, and $H_w \in \mathbb{R}^{mT \times mT}$ is the multi-task kernel matrix corresponding to the multi-task kernel $h_w \; \forall \; w \in \mathcal{V}$. The kernel h_w is defined as follows

$$h_w(\mathbf{x}_{t_1 i}, \mathbf{x}_{t_2 j}) = k_w(\mathbf{x}_{t_1 i}, \mathbf{x}_{t_2 j}) B(t_1, t_2) \tag{5.8}$$

where B is a $T \times T$ matrix. $B = I$ (identity matrix) when the regularizer (5.4) is employed in (5.3). Alternatively, $B = I + \mathbf{1}\mathbf{1}^\top/\mu$ (here $\mathbf{1}$ is a $T \times 1$ vector with entries as unity) in the case when the regularizer (5.5) is employed. The prediction function of the task t_1 is given by

$$F_{t_1}(\mathbf{x}_{t_1 j}) = \sum_{t_2=1}^{T} \sum_{i=1}^{m} \bar{\alpha}_{t_2 i} y_{t_2 i} \left(\sum_{w \in \mathcal{V}} \delta_w(\bar{\gamma}, \bar{\lambda}) k_w(\mathbf{x}_{t_1 i}, \mathbf{x}_{t_2 j}) B(t_1, t_2) \right)$$

where $(\bar{\gamma}, \bar{\lambda}, \bar{\alpha})$ is an optimal solution of (5.7).

Proof. The proof follows from the representer theorem [113]. Also refer to appendix A.1.3. □

This lemma shows that the gHKL$_{\text{MT}}$ formulation essentially constructs the same prediction function as an SVM with the effective multi-task kernel as: $h = \sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) h_w$. Similarly, in the case of the gHKL, the effective kernel is $k = \sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) k_w$ (since the terms T and B are unity). Here, as well as in the rest of the chapter, we employ the symbols h and H for the multi-task kernel and the corresponding kernel matrix respectively.

The multi-task kernel (5.8) consists of two terms: the first term corresponds to the similarity between two instances $\mathbf{x}_{t_1 i}$ and $\mathbf{x}_{t_2 j}$ in the feature space induced by kernel k_w. The second term corresponds to the correlation between the tasks t_1 and t_2. In the case of the regularizer (5.4), the matrix B simplifies to: $B(t_1, t_2) = 1$ if $t_1 == t_2$ and $B(t_1, t_2) = 0$ if $t_1 \neq t_2$, thereby making the kernel matrices H_w ($w \in \mathcal{V}$) block diagonal. Hence, the gHKL$_{\text{MT}}$ regularizer based on (5.4) promotes simultaneous sparsity in kernel selection among the tasks, without enforcing any additional correlations among the tasks.

In general, any $T \times T$ positive semi-definite matrix may be employed as B to model generic correlations among tasks. The multi-task kernel given by (5.8) will still remain a valid kernel [115, 3]. The matrix B is sometimes referred to as the output kernel in the setting of learning vector-valued functions. It is usually constructed from the prior domain knowledge.

We now discuss the nature of the optimal solution of (5.7). Most of the kernel weights $\delta_w(\gamma, \lambda)$ are zero at optimality of (5.7): $\delta_w(\gamma, \lambda) = 0$ whenever $\gamma_v = 0$ or $\lambda_{vw} = 0$ for any

Algorithm 1 Active Set Algorithm - Outline

```
Input: Training data $\mathcal{D}$, the kernels $(k_v)$ embedded on the DAG $(\mathcal{V})$, the $T \times T$ matrix $B$ that models task correlations and tolerance $\epsilon$.
Initialize the active set $\mathcal{W}$ with $sources(\mathcal{V})$.
Compute $\gamma, \lambda, \alpha$ by solving (5.9)
while Optimal solution for (5.7) is NOT obtained do
    Add some nodes to $\mathcal{W}$
    Recompute $\eta, \alpha$ by solving (5.9)
end while
Output: $\mathcal{W}, \gamma, \lambda, \alpha$
```

$v \in A(w)$. The vector γ is sparse due to ℓ_1-norm constraint in (5.7). In addition, $\rho \to 1 \Rightarrow \hat{\rho} \to 1$. Hence the vectors $\lambda_v \; \forall \; v \in \mathcal{V}$ get close to becoming sparse as $\rho \to 1$ due to the $\ell_{\hat{\rho}}$-norm constraint in (5.7). The superimposition of these two phenomenon leads to a flexible[3] sparsity pattern in kernel selection. In particular, it may happen that $\exists w \in \mathcal{V}, v \in A(w) \mid \delta_w(\gamma, \lambda) > \delta_v(\gamma, \lambda)$. This instance of weighting scheme is not possible in the HKL.

Note that the problem (5.7) remains the same whether solved with the original set of variables (*i.e.*, γ, λ) or when solved with only those $\gamma_v \neq 0$ and $\lambda_{vw} \neq 0$ at optimality (refer appendix A.1.4 for details). However the computational effort required in the latter case can be far lower since it involves low number of variables. This motivates us to explore an active set algorithm, which is similar in spirit to that in [11].

An outline of the proposed active set algorithm is presented in Algorithm 1. The algorithm starts with an initial guess for the set of γ_w that are non-zero at optimality for (5.7). This set is called the active set and is denoted by $\mathcal{W}$. Since the kernel weight $\delta_w(\gamma, \lambda) = 0$ whenever $\gamma_v = 0$ for any $v \in A(w)$, the active set must contain $sources(\mathcal{V})$, else the problem has a trivial solution. Hence, the active set is initialized with $sources(\mathcal{V})$. At each iteration, the problem

[3]HKL dual formulation [10] is a special case of (5.7) with $\rho = 2$, $T = 1$ and $B = 1$. When $\rho = 2$, $\hat{\rho} = \infty$. This implies $\lambda_{vw} = 1 \; \forall \; v \in A(w), \; w \in \mathcal{V}$ at optimality, resulting in the weight bias towards kernels embedded in the ancestor nodes and restricted sparsity pattern in kernel selection

(5.7) with variables restricted to those in $\mathcal{W}$ is solved:

$$\min_{\gamma \in \Delta_{|\mathcal{V} \cap \mathcal{W}|,1}} \quad \min_{\lambda_v \in \Delta_{|D(v) \cap hull(\mathcal{W})|,\hat{\rho}} \forall v \in \mathcal{W}} \quad \max_{\alpha_t \in S(\mathbf{y}_t, C) \forall t} G_{\mathcal{W}}(\gamma, \lambda, \alpha) \tag{5.9}$$

where

$$G_{\mathcal{W}}(\gamma, \lambda, \alpha) = \mathbf{1}^\top \alpha - \frac{1}{2} \alpha^\top \mathbf{Y} \left(\sum_{w \in \mathcal{W}} \delta_w(\gamma, \lambda) H_w \right) \mathbf{Y} \alpha$$

Note that $\rho = 2 \Rightarrow \lambda_{vw} = 1 \ \forall \ v \in A(w), \ w \in \mathcal{W}$ at optimality in (5.9). Hence for $\rho = 2$, the minimization problem in (5.9) can be efficiently solved using a projected gradient method [108, 10]. However, as established in [90], projection onto the kind of feasibility set in the minimization problem in (5.9) is computationally challenging for $\rho \in (1, 2)$. Hence, we wish to re-write this problem in a relatively simpler form that can be solved efficiently. To this end, we present the following important theorem:

Theorem 5.2.3. *The following is a dual of (5.3) considered with the hinge loss function, and the objectives of (5.3) (with the hinge loss), (5.7) and (5.10) are equal at optimality:*

$$\min_{\eta \in \Delta_{|\mathcal{V}|,1}} \quad g(\eta) \tag{5.10}$$

where $g(\eta)$ is the optimal objective value of the following convex problem:

$$\max_{\alpha_t \in S(\mathbf{y}_t, C) \forall t} \quad \mathbf{1}^\top \alpha - \frac{1}{2} \left(\sum_{w \in \mathcal{V}} \zeta_w(\eta) \left(\alpha^\top \mathbf{Y} H_w \mathbf{Y} \alpha \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}} \tag{5.11}$$

where $\zeta_w(\eta) = \left(\sum_{v \in A(w)} d_v^\rho \eta_v^{1-\rho} \right)^{\frac{1}{1-\rho}}$, $\alpha = [\alpha_1^\top, \ldots, \alpha_T^\top]^\top$, $S(\mathbf{y}_t, C) = \{\beta \in \mathbb{R}^m \mid 0 \leq \beta \leq C, \ \sum_{i=1}^m y_{ti} \beta_i = 0\}$, $\mathbf{y}_t = [y_{t1}, \ldots, y_{tm}]^\top \ \forall \ t$, $\mathbf{Y}$ is the diagonal matrix corresponding to the vector $[\mathbf{y}_1^\top, \ldots, \mathbf{y}_T^\top]^\top$, $\mathbf{1}$ is a $mT \times 1$ vector with entries as unity, $\Delta_{n,1} \equiv \{\eta \in \mathbb{R}^n \mid \eta \geq 0, \sum_{i=1}^n \eta_i \leq 1\}$, $\bar{\rho} = \frac{\hat{\rho}}{\hat{\rho}-1}$, $\hat{\rho} = \frac{\rho}{2-\rho}$, and $H_w \in \mathbb{R}^{mT \times mT}$ is the multi-task kernel matrix corresponding to the multi-task kernel (5.8).

The key idea in the proof of the above theorem is to eliminate the λ variables and the details are presented in appendix A.1.5. The expression for the prediction function F, in terms of the variables η and α, is provided in appendix A.1.9.

This theorem provides some key insights: firstly, we have that (5.10) is essentially a ℓ_1-norm regularized problem and hence it is expected that most of the η will be zero at optimality.

Since $(\eta_v = 0) \Rightarrow (\zeta_w(\eta) = 0 \; \forall \; w \in D(v))$, it follows that most of the nodes in $\mathcal{V}$ will not contribute in the optimization problems (5.10) and (5.11). Secondly, in a single task learning setting $(T = 1)$, the problem in (5.11) is equivalent to the $\ell_{\hat{\rho}}$-norm MKL dual problem [81] with the base kernels as $(\zeta_v(\eta))^{\frac{1}{\hat{\rho}}} k_v \; \forall \; v \in \mathcal{V} \ni \zeta_v(\eta) \neq 0$. The optimization problem (5.11) essentially learns an effective kernel of the form $h = \sum_{v \in \mathcal{V}} \theta_v \left(\zeta_v(\eta)\right)^{\frac{1}{\hat{\rho}}} h_v$, where the θ are intermediate optimization variables constrained to lie in the feasibility set $\Delta_{|\mathcal{V}|,\hat{\rho}}$. The expression of θ in terms of the variables η and α is provided in appendix A.1.9.

The θ influences the nature of the effective kernel h in two important ways: i) it follows from the expression of θ that

$$\theta_v \left(\zeta_v(\eta)\right)^{\frac{1}{\hat{\rho}}} \propto \zeta_v(\eta) \left(\alpha^\top \mathbf{Y} H_v \mathbf{Y} \alpha\right)^{\frac{1}{(\hat{\rho}-1)}}$$

The above relation implies that the weight of the kernel h_v in the DAG $\mathcal{V}$ is not only dependent on the position[4] of the node v, but also on the suitability of the kernel h_v to the problem at hand. This helps in mitigating the kernel weight bias in favour of the nodes towards the top of the DAG from the gHKL$_{\text{MT}}$, but which is present in the HKL, and ii) as $\rho \to 1$ (and hence as $\hat{\rho} \to 1$), the optimal θ get close to becoming sparse [119, 104]. This superimposed with the sparsity of η promotes a more flexible sparsity pattern in kernel selection that HKL, especially when $\rho \to 1$.

Due to the sparsity of η at optimality, we propose to solve problem (5.10) by an active set algorithm, based on the outline presented in Algorithm 1. We discuss it in detail in the next section.

5.3 Optimization Algorithm

In order to formalize the active set algorithm, we need: i) an efficient algorithm for solving problem (5.10) restricted to the active set, *i.e.*, $\mathcal{V}$ restricted to $\mathcal{W}$, ii) a condition for verifying whether a candidate solution is optimal with respect to the problem (5.10), and iii) a procedure for building/improving the active set after each iteration.

[4]Similar to the δ_v function in the HKL (2.6), it follows from the definition of ζ_v that $\zeta_v(\eta) \geq \zeta_w(\eta) \; \forall \; w \in D(v)$ (strict inequality holds if $\zeta_w(\eta) > 0$).

Algorithm 2 Mirror Descent Algorithm for solving (5.10) restricted to active set $\mathcal{W}$

```
Input: Kernel matrices $K_w$ $(w \in \mathcal{W})$ and the regularization parameter $C$
Initialize $\eta_{\mathcal{W}}$ $(w \in \mathcal{W})$ such that $\eta_{\mathcal{W}} \in \Delta_{|\mathcal{W}|,1}$ (warm-start may be used)
Iteration number: $i = 0$
while convergence criterion is not met[5] do
    $i = i + 1$
    Compute $\zeta_w(\eta_{\mathcal{W}})$ $\forall$ $w \in \mathcal{W}$ (Theorem 5.2.3)
    Compute $\alpha_{\mathcal{W}}$ (5.11) using $\ell_{\hat{\rho}}$-norm MKL algorithm with kernels as $\left((\zeta_w(\eta_{\mathcal{W}}))^{\frac{1}{\hat{\rho}}} H_w\right)_{w \in \mathcal{W}}$
    Compute $\nabla g(\eta_{\mathcal{W}})$ as in (A.9)
    Compute step size $s = \sqrt{\log(|\mathcal{W}|)/i \cdot \|\nabla g(\eta_{\mathcal{W}})\|_\infty^2}$
    Compute $\eta_w = \exp\left(1 + \log(\eta_w) - s \cdot \nabla g(\eta_{\mathcal{W}})_w\right)$ $\forall$ $w \in \mathcal{W}$
    Normalize $\eta_w = \frac{\eta_w}{\sum_{v \in \mathcal{W}} \eta_v}$ $\forall$ $w \in \mathcal{W}$
end while
Output: $\eta_{\mathcal{W}}, \alpha_{\mathcal{W}}$
```

We begin with the first. We propose to solve the optimization problem (5.10) restricted to $\mathcal{W}$ using the mirror descent algorithm [99]. Mirror descent algorithm is known to efficiently solve convex programs with Lipschitz continuous and sub-differentiable objectives constrained over a convex compact set. It achieves a near-optimal convergence rate whenever the feasibility set is a simplex (which is true in our case). The mirror descent algorithm is close in spirit to the projected gradient descent algorithm and hence assumes that an oracle for computing the gradient of the objective is available.

Following the common practice of smoothing [10], in the rest of the chapter, we employ $\zeta_w((1-\varepsilon)\eta + \frac{\varepsilon}{|V|})$ instead[6] of $\zeta_w(\eta)$ in (5.11) with $\varepsilon > 0$. The following theorem establishes the applicability of mirror descent for solving (5.10):

Theorem 5.3.1. *The function $g(\eta)$ given by (5.11) is convex. Also, the expression for the i^{th} entry in the gradient $(\nabla g(\eta))_i$ is given in (A.9). If all the eigen-values of the gram-matrices H_w*

[5]Relative objective gap between two successive iteration being less than a given tolerance ϵ is taken to be the convergence criterion. Objective here is the value of $g(\eta_{\mathcal{W}})$, calculated after $\ell_{\hat{\rho}}$-norm MKL step.

[6]Note that this is equivalent to smoothing the regularizer $\Omega_T(f)$ while preserving its sparsity inducing properties [10].

Algorithm 3 Active Set Algorithm

```
Input: Training data $\mathcal{D}$, the kernels embedded on the DAG and tolerance $\epsilon$
Initialize the active set $\mathcal{W}$ with $sources(\mathcal{V})$
Compute $\eta, \alpha$ by solving (5.10) using Algorithm 2
while sufficient condition for optimality (5.12) is not met do
    Add those nodes to $\mathcal{W}$ that violate (5.12)
    Recompute $\eta, \alpha$ by solving (5.10) using Algorithm 2
end while
Output: $\mathcal{W}, \eta, \alpha$
```

are finite and non-zero, then g is Lipschitz continuous.

Proof of the above theorem is technical and is provided in appendix A.1.6.

Algorithm 2 summarizes the proposed mirror descent based algorithm for solving (5.10) with $\mathcal{V}$ restricted to $\mathcal{W}$. One of its steps involve computing $\nabla g(\eta_{\mathcal{W}})$ (expression provided in (A.9)), which in turn requires solving (5.11). As noted before, (5.11) is similar to the $\ell_{\hat{\rho}}$-norm MKL problem [81] but with a different feasibility set for the optimization variables α. Hence, (5.11) can be solved by employing a modified cutting planes algorithm [81] or a modified sequential minimal optimization (SMO) algorithm [105, 128]. Empirically, we observed the SMO based algorithm to be much faster than the cutting planes algorithm for gHKL$_{\text{MT}}$ (and gHKL) with SVM loss functions. In the special case $\rho = 2, T = 1$ and $B = 1$, (5.11) is simply a regular SVM problem.

Now we turn our attention to the second requirement of the active set algorithm: a condition to verify the optimality of a candidate solution. We present the following theorem that provides a sufficient condition for verifying optimality of a candidate solution.

Theorem 5.3.2. *Suppose the active set $\mathcal{W}$ is such that $\mathcal{W} = hull(\mathcal{W})$. Let $(\eta_{\mathcal{W}}, \alpha_{\mathcal{W}})$ be a $\epsilon_{\mathcal{W}}$-approximate optimal solution for (5.10) with $\mathcal{V}$ restricted to $\mathcal{W}$ returned by Algorithm (2). Then it is an optimal solution for (5.10) with a duality gap less than ϵ if:*

$$\max_{u \in sources(\mathcal{W}^c)} \alpha_{\mathcal{W}}^\top \mathbf{Y} \mathcal{K}_u \mathbf{Y} \alpha_{\mathcal{W}} \leq \left(\sum_{w \in \mathcal{W}} \zeta_w(\eta_{\mathcal{W}}) \left(\alpha_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y} \alpha_{\mathcal{W}} \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}} + 2(\epsilon - \epsilon_{\mathcal{W}}) \qquad (5.12)$$

where $\mathcal{K}_u = \sum_{w \in D(u)} \frac{H_w}{\left(\sum_{v \in A(w) \cap D(u)} d_v \right)^2}$.

Proof is provided in appendix A.1.7 and closely follows that for the case of HKL [11].

The summary of the proposed mirror descent based active set algorithm is presented in Algorithm 3. At each iteration, Algorithm (3) verifies optimality of the current iterate by verifying the condition in (5.12). In case the current iterate does not satisfy this condition, then the nodes in $sources(\mathcal{W}^c)$ that violate the condition (5.12) are included in the active set[7]. This takes care of the third requirement of the active set algorithm. The algorithm terminates if the condition (5.12) is satisfied by the iterate.

In the following, an estimate of the computational complexity of the active set algorithm is presented. Let W be the final active set size. The optimization problem (5.10) restricted to $\mathcal{W}$ needs to be solved at most W times, assuming the worst case scenario of adding one node per active set iteration. Each such run of the mirror descent algorithm requires at most $O(\log(W))$ iterations [99]. A conservative time complexity estimate for computing the gradient $\nabla g(\eta_{\mathcal{W}})$ (by solving the variant of the $\ell_{\hat{\rho}}$-norm MKL problem (5.11)) is $O(m^3T^3W^2)$. This amounts to $O(m^3T^3W^3\log(W))$. As for the computational cost of the sufficient condition, let z denote the maximum out-degree of a node in $\mathcal{G}$, *i.e.*, z is an upper-bound on the the maximum number of children of any node in $\mathcal{G}$. Then the size of $sources(\mathcal{W}^c)$ is upper-bounded by Wz. Hence, a total of $O(\omega m^2T^2Wz)$ operations is required for evaluating $\mathcal{K}$ in (5.12), where ω is the complexity of computing a single entry in any $\mathcal{K}$. In all the pragmatic examples of kernels and corresponding DAGs provided by [11], ω is polynomial in the training set dimensions. Moreover, caching of $\mathcal{K}$ usually renders ω to be a constant [10]. Further, the total cost of the quadratic computation in (5.12) is $O(m^2T^2W^2z)$. Thus the overall computational complexity is: $O(m^3T^3W^3\log(W)+\omega m^2T^2Wz+m^2T^2W^2z)$. More importantly, because the sufficiency condition for optimality given by Theorem 5.3.2 is independent of ρ, we have the following result:

Corollary 5.3.3. *In a given input setting, the HKL algorithm converges in time polynomial in size of the active set and the training set dimensions if and only if the proposed mirror descent based active set algorithm, i.e., the gHKL*$_{\text{MT}}$ *algorithm, has a polynomial time convergence in terms of the active set and training set sizes.*

Proof is provided in Appendix A.1.10.

[7]It is easy to see that with this update scheme, $\mathcal{W}$ is always equal to $hull(\mathcal{W})$, as required in Theorem 5.3.2.

In the next section, we present an application of the gHKL formulation that illustrate the benefits of the proposed generalizations over the HKL.

5.4 Rule Ensemble Learning

In this section, we propose a solution to the problem of learning an ensemble of decision rules, formally known as Rule Ensemble Learning (REL) [36], employing the gHKL and gHKL_{MT} formulations. For the sake of simplicity, we only discuss the single task REL setting in this section, *i.e.* REL as an application of the gHKL formulation. Similar ideas can be applied to perform REL in multi-task learning setting, by employing the gHKL_{MT} formulation. In fact, we present empirical results of REL in both single and multiple task learning settings in Section 5.5. We begin with a brief introduction to REL.

If-then decision rules [107] are one of the most expressive and human readable representations for learned hypotheses. It is a simple logical pattern of the form: IF *condition* THEN *decision*. The *condition* consists of a conjunction of a small number of simple boolean statements (propositions) concerning the values of the individual input variables while the *decision* specifies a value of the function being learned. An instance of a decision rule from Quinlan's playtennis example [106] is:

IF *HUMIDITY==normal* AND *WIND==weak* THEN *PlayTennis==yes*

The dominant paradigm for induction of rule sets, in the form of decision list (DL) models for classification [107, 97, 35], has been a greedy *sequential covering* procedure.

REL is a general approach that treats decision rules as base classifiers in an ensemble. This is in contrast to the more restrictive DL models that are disjunctive sets of rules and which use only one in the set for each prediction. As pointed out in [36], boosted rule ensembles are in fact simpler, better-understood formally than other state-of-the-art rule learners and also produce comparable predictive accuracy.

REL approaches like SLIPPER [36], LRI [134], RuleFit [51], ENDER/MLRules [45, 46] have additionally addressed the problem of learning a compact set of rules that generalize well in order to maintain their readability. Further, a number of rule learners like RuleFit, LRI encourage shorter rules (*i.e.*, fewer conjunctions in the condition part of the rule) or rules with a restricted number of conjunctions, again for purposes of interpretability. We build upon this and define our REL problem as that of learning a small set of simple rules and their weights that leads to a good generalization over new and unseen data. The next section introduces the notations and the setup in context of REL.

Notations and Setup

Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ be the training data described using p basic propositions, *i.e.*, $\mathbf{x}_i \in \{0,1\}^p$. In case the input features are not boolean, then such propositions can be derived using logical operators such as $==, \neq, \leq, \geq$ etc. over the input features [refer 51, 45, for details]. Let $\mathcal{V}$ be an index-set for all possible conjunctions with the p basic propositions and let $\phi_v : \mathbb{R}^n \mapsto \{0,1\}$ denote the v^{th} conjunction in $\mathcal{V}$. Let $f_v \in \mathbb{R}$ denote the weight for the conjunction ϕ_v. Then, the rule ensemble to be learnt is the weighted combination of these conjunctive rules: $F(\mathbf{x}) = \sum_{v \in \mathcal{V}} f_v \phi_v(\mathbf{x}) - b$, where perhaps many of the weights $f_v = 0$.

One way to learn the weights is by performing a ℓ_1-norm regularized risk minimization in order to select few promising conjunctive rules [51, 45, 46]. However, to the best of our knowledge, rule ensemble learners that identify the need for sparse f, either approximate such a regularized solution using strategies such as shrinkage (Rulefit, ENDER/MLRules) or resort to post-pruning (SLIPPER). This is because the size of the minimization problem is exponential in the number of basic propositions and hence the problem becomes computationally intractable with even moderately sized datasets. Secondly, conjunctive rules involving large number of propositions might be selected. However, such conjunctions adversely effect the interpretability. Here we present an approach based on the gHKL framework that addresses these issues.

We begin by noting that $\langle \mathcal{V}, \subseteq \rangle$ is a subset-lattice; hereafter this will be referred to as the *conjunction lattice*. Here $\forall\, v_1, v_2 \in \mathcal{V},\ v_1 \subseteq v_2$ *iff* the set of propositions in the v_1 conjunction is a subset of those in the v_2 conjunction. As an example, (*HUMIDITY==normal*) is considered to be a subset of (*HUMIDITY==normal* AND *WIND==weak*). The top node of this lattice is a node with no conjunctions and is also $sources(\mathcal{V})$. Its children, the second level nodes, are all the basic propositions, p in number. The third level nodes, children of these basic propositions, are the conjunctions of length two and so on. The bottom node at $(p+1)^{th}$ level is the conjunc-

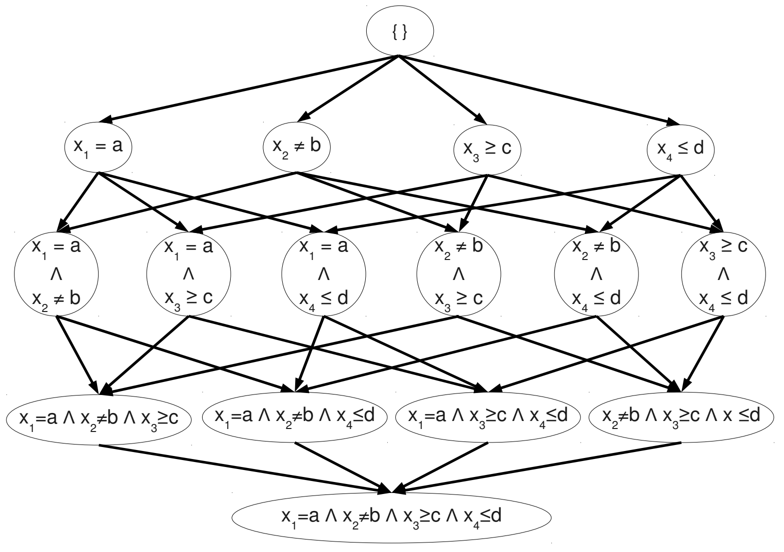

Figure 5.1: A lattice of conjunction with four basic propositions: $(x_1 = a)$, $(x_2 \neq b)$, $(x_3 \geq c)$ and $(x_4 \leq d)$. Here x_1, x_2, x_3 and x_4 belong to the input space.

tion of all basic propositions. Figure (5.1) shows such a lattice with all it nodes when $p = 4$.

We now discuss how the proposed gHKL regularizer (5.2) provides an efficient and optimal solution to a regularized empirical risk minimization formulation for REL.

Rule Ensemble Learning using gHKL

The key idea is to employ the gHKL formulation (5.2) with the DAG as the conjunction lattice and the kernels as $k_v(\mathbf{x}_i, \mathbf{x}_j) = \phi_v(\mathbf{x}_i)\phi_v(\mathbf{x}_j)$ for performing the task of REL. Note that with such a set-up, the ℓ_1/ℓ_ρ block-norm regularizer in gHKL, $\Omega_S(f) = \sum_{v \in \mathcal{V}} d_v \|f_{D(v)}\|_\rho$, implies: 1) for most $v \in \mathcal{V}$, $f_v = 0$, and 2) for most $v \in \mathcal{V}$, $f_w = 0 \; \forall \; w \in D(v)$. In the context of the REL problem, the former statement is same as saying: selection of a compact set of conjunctions is promoted, while the second reads as: selection of conjunctive rules with small number of propositions is promoted. Thus, the gHKL formulation constructs a compact ensemble of simple conjunctive rules. In addition, we choose $d_v = a^{|S_v|}$ $(a > 1)$, where S_v is the set of basic propositions involved in the conjunction ϕ_v. Such a choice further encourages selection of short conjunctions and leads to the following elegant computational result:

Theorem 5.4.1. *The complexity of the proposed gHKL algorithm in solving the REL problem,* *with the DAG, the base kernels and the parameters d_v as defined above, is polynomial in the size of the active set and the training set dimensions. In particular, if the final active set size is W, then its complexity is given by $O(m^3 W^3 \log(W) + m^2 W^2 p)$.*

Proof is provided in appendix A.1.11.

We end this section by noting the advantage of the generic regularizer in the gHKL formulation over the that in the HKL formulation in the context of REL applications. Recall that the sparsity pattern allowed by the HKL formulation has the following consequence: a conjunction is selected only after selecting all the conjunctions which are subsets of it. This, particularly in the context of REL, is psycho-visually redundant, because a rule with n propositional statements, if included in the result, will necessarily entail the inclusion of $(2^n - 1)$ more general rules in the result. This violates the important requirement for a small set [51, 45, 46] of human-readable rules. The gHKL regularizer, with $\rho \in (1, 2)$, alleviates this restriction by promoting additional sparsity in selecting the conjunctions. We empirically evaluate the proposed gHKL based solution of the REL problem in the next section.

5.5 Experiments

In this section, we report the results of simulation in REL on several benchmark binary and multiclass classification datasets from the UCI repository [50]. The goal is to compare various rule ensemble learners on the basis of: (a) generalization, which is measured by the predictive performance on unseen test data, and (b) ability to provide compact set of short (simple) rules to facilitate their readability and interpretability [51, 46, 36]. This is judged using i) average number of rules selected, and ii) average number of propositions employed per rule. The following methods were compared.

RuleFit: Rule ensemble algorithm proposed by [51]. All the parameters were set to the default values mentioned by the authors. In particular, the model was set in the mixed linear-rule mode, average tree size was set 4 and maximum number of trees was kept as 500. The same configuration was also used by [45, 46] in their simulations. This REL system cannot handle multi-class datasets and hence is limited to the simulations on binary classification datasets. Its code is available at `www-stat.stanford.edu/~jhf/R-RuleFit.html`.

SLI: The SLIPPER algorithm proposed by [36]. Following [45, 46], all parameters were set to their defaults. We retained the internal cross-validation for selecting the optimal number of rules.

ENDER: State-of-the-art rule ensemble algorithm [46]. For classification setting, **ENDER** is same as MLRules [45]. The parameters were set to the default values suggested by the authors. The second order heuristic was used for minimization. Its code is available at `www.cs.put.poznan.pl/wkotlowski`.

gHKL$_\rho$: The proposed gHKL based REL formulation for binary classification problem. We considered three different values of ρ: 2, 1.5 and 1.1. Note that for binary classification, $\rho = 2$ renders the HKL formulation [11]. In each case, a 3-fold cross validation procedure was employed to tune the C parameter with values in $\{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$. As mentioned earlier, the parameters $d_v = 2^{|v|}$.

gHKL$_{\mathbf{MT}-\rho}$: The proposed gHKL$_{\text{MT}}$ based REL formulation for multiclass classification problem. For each class, a 1-vs-rest binary classification task is created. Since we did not have any prior knowledge about the correlation among the classes in the datasets, we employed the multi-task regularizer (5.4) in the gHKL$_{\text{MT}}$ primal formulation (5.3). Correspondingly, the B matrix in the gHKL$_{\text{MT}}$ dual formulation (5.10) is set to the identity matrix. We considered three different values of ρ: 2, 1.5 and 1.1. Its parameters and cross validation details are same as that of **gHKL$_\rho$**. The codes of **gHKL$_\rho$** and **gHKL$_{\mathbf{MT}-\rho}$** are available at `http://www.cse.iitb.ac.in/~pratik.j/ghkl`.

Note that the above methods differ in the way they control the number of rules (M) in the ensemble. In the case of **gHKL$_\rho$** (**gHKL$_{\mathbf{MT}-\rho}$**), M depends on the parameters: ρ, C and d_v. **SLI** has a parameter for maximum number of rules M_{max} and M is decided via a internal cross-validation such that $M \leq M_{max}$. For the sake of fairness in comparison with **gHKL$_\rho$**, we set $M_{max} = \max(M_{1.5}, M_{1.1})$, where M_ρ is the average number of rules obtained with **gHKL$_\rho$** (**gHKL$_{\mathbf{MT}-\rho}$**). **ENDER** has an explicit parameter for the number of rules, which is also set

Dataset	# of Examples	Bias	p	$\|\mathcal{V}\|$	Dataset	# of Examples	Bias	p	$\|\mathcal{V}\|$
TIC-TAC-TOE	958	1.89	54	$\approx 10^{16}$	HEARTSTAT	270	0.8	76	$\approx 10^{22}$
B-CANCER-W	699	0.53	72	$\approx 10^{21}$	MONK-3	554	1.08	30	$\approx 10^{9}$
DIABETES	768	0.54	64	$\approx 10^{19}$	VOTE	232	0.87	32	$\approx 10^{9}$
HABERMAN	306	0.36	40	$\approx 10^{12}$	B-CANCER	277	0.41	76	$\approx 10^{22}$
HEARTC	296	0.85	78	$\approx 10^{23}$	MAM. MASS	829	0.94	46	$\approx 10^{13}$
BLOOD TRANS	748	3.20	32	$\approx 10^{9}$	LIVER	345	1.38	48	$\approx 10^{14}$

Table 5.1: Datasets used for binary REL classification. Bias denotes the ratio of $\#$ of $+$ve and $-$ve instances. p denotes the number of basic propositions. For each numerical input feature, 8 basic propositions were derived.

to $\max(M_{1.5}, M_{1.1})$. In case of **RuleFit**, the number of rules in the ensemble is determined internally and is not changed by us.

Binary Classification in REL

This section summarizes the simulations with binary classification datasets. Table 5.1 provides the details of the binary classification datasets. For every dataset, we created 10 random train-test splits with 10% train data[8]. Since many datasets were highly unbalanced, we report[9] the average F1-score (with standard deviation). The results are presented in Table 5.2. The best result, in terms of the average F1-score, for each dataset is highlighted. Additionally if the best result achieves a statistically significant improvement over its nearest competitor, it is marked with a '*'. Statistical significance test is performed using the paired t-test at 99% confidence. We also report the average number of rules produced per split and the average length of the rules, specified below each F1-score as: (average number of rules, average length of rules). As discussed earlier, it is desirable that REL algorithms achieve high F1-score with a compact set of simple rules.

We can observe from Table 5.2 that **gHKL**$_\rho$ obtains better generalization performance than state-of-the-art in most of the datasets with the additional advantage of having rules with smaller number of conjunctions. In fact, when averaged over the datasets, **gHKL**$_{1.1}$ and **gHKL**$_{1.5}$ output the shortest rules among all the methods. **gHKL**$_{1.1}$ obtains statistically significant performance in TIC-TAC-TOE, BLOOD TRANS and B-CANCER datasets. Though the generalization ob-

[8]For monk-3, a single train-test split of $122 - 432$ instances respectively was given in UCI repository itself.

[9]Table A.1 in appendix A.1.12 reports the average AUC.

Dataset	RuleFit	SLI	ENDER	gHKL$_\rho$		
				$\rho = 2$	$\rho = 1.5$	$\rho = 1.1$
TIC-TAC-TOE	0.517 ± 0.092	0.665 ± 0.053	0.668 ± 0.032	0.889 ± 0.093	0.897 ± 0.093	$\mathbf{0.905 \pm 0.096}^*$
	(57.7, 2.74)	(10.3, 1.96)	(187, 3.17)	(161.7, 1.715)	(186.6, 1.761)	(157.6, 1.717)
B-CANCER-W	0.879 ± 0.025	$\mathbf{0.928 \pm 0.018}$	0.9 ± 0.041	0.923 ± 0.032	0.924 ± 0.032	0.925 ± 0.032
	(17.5, 2.027)	(4.4, 1.15)	(21, 1.56)	(30.9, 1)	(20, 1.033)	(20.4, 1.023)
DIABETES	0.428 ± 0.052	0.659 ± 0.027	0.656 ± 0.027	0.661 ± 0.018	$\mathbf{0.663 \pm 0.017}$	0.661 ± 0.023
	(32.9, 2.66)	(4.9, 1.42)	(74.0, 2.65)	(83.2, 1.305)	(73.2, 1.172)	(62.6, 1.27)
HABERMAN	0.175 ± 0.079	0.483 ± 0.057	0.474 ± 0.057	$\mathbf{0.523 \pm 0.062}$	0.521 ± 0.06	0.521 ± 0.06
	(7.5, 1)	(2.1, 1)	(52, 3.59)	(112.1, 1.366)	(51.2, 1.235)	(17.1, 1.142)
HEARTC	0.581 ± 0.047	0.727 ± 0.05	0.724 ± 0.032	$\mathbf{0.743 \pm 0.038}$	0.735 ± 0.058	0.736 ± 0.055
	(8.8, 1)	(3.2, 1.23)	(32, 2.05)	(46.7, 1.056)	(23.9, 1)	(32, 1.086)
BLOOD TRANS	0.163 ± 0.088	0.476 ± 0.057	0.433 ± 0.0	0.586 ± 0.029	0.587 ± 0.028	$\mathbf{0.588 \pm 0.027}^*$
	(40.7, 2.26)	(2.0, 1)	(63, 1.97)	(229.7, 1.981)	(62.8, 1.789)	(19, 1.294)
HEARTSTAT	0.582 ± 0.04	0.721 ± 0.065	0.713 ± 0.055	$\mathbf{0.747 \pm 0.031}$	0.746 ± 0.028	$\mathbf{0.747 \pm 0.028}$
	(9.3, 1)	(3.5, 1.07)	(25, 2.02)	(34.7, 1.023)	(25, 1.017)	(24.4, 1.027)
MONK-3	0.947	0.802	**0.972**	**0.972**	**0.972**	**0.972**
	(52, 2.88)	(1, 3)	(93, 1.96)	(200, 2.065)	(93, 1.839)	(7, 1.429)
VOTE	0.913 ± 0.047	0.935 ± 0.055	$\mathbf{0.951 \pm 0.035}$	0.93 ± 0.042	0.929 ± 0.043	0.934 ± 0.038
	(2.7, 1)	(1.3, 1.15)	(9, 1.07)	(39, 1.106)	(8.2, 1)	(6.4, 1)
B-CANCER	0.254 ± 0.089	0.476 ± 0.086	0.452 ± 0.079	0.565 ± 0.059	0.563 ± 0.061	$\mathbf{0.569 \pm 0.063}^*$
	(8.1, 1)	(1.2, 1)	(31, 2.93)	(39.6, 1.154)	(30.2, 1.068)	(29.4, 1.167)
MAM. MASS	0.668 ± 0.032	0.808 ± 0.022	$\mathbf{0.816 \pm 0.018}$	0.796 ± 0.026	0.796 ± 0.026	0.797 ± 0.024
	(26.4, 2.68)	(5.3, 1.43)	(48, 2.53)	(92.2, 1.274)	(47.6, 1.237)	(40.5, 1.252)
LIVER	0.357 ± 0.016	0.445 ± 0.083	0.563 ± 0.058	0.594 ± 0.046	$\mathbf{0.595 \pm 0.048}$	0.588 ± 0.049
	(10, 1)	(1.5, 1)	(59, 2.35)	(242.5, 1.422)	(58.2, 1.319)	(45.7, 1.358)

Table 5.2: Results on binary REL classification. F1-score along with standard deviation, the number of rules and the length of rules averaged over 10 splits are shown.

tained by **gHKL**$_2$ (HKL), **gHKL**$_{1.5}$ and **gHKL**$_{1.1}$ are similar, the number of rules selected by **gHKL**$_2$ is generally higher than **gHKL**$_{1.1}$ (by 5 to 25 times in some cases), hampering its interpretability.

Dataset	# of Examples	c	p	$\lvert\mathcal{V}\rvert$	Dataset	# of Examples	c	p	$\lvert\mathcal{V}\rvert$
BALANCE	625	3	32	$\approx 10^9$	IRIS	150	3	50	$\approx 10^{15}$
CAR	1728	4	42	$\approx 10^{12}$	LYMPH	146	3	86	$\approx 10^{25}$
C.M.C.	1473	3	54	$\approx 10^{16}$	T.A.E.	151	3	114	$\approx 10^{34}$
ECOLI	332	6	42	$\approx 10^{12}$	YEAST	1484	10	54	$\approx 10^{16}$
GLASS	214	6	72	$\approx 10^{21}$	ZOO	101	7	42	$\approx 10^{12}$

Table 5.3: Datasets used for multiclass REL classification. c denotes the number of classes.

Dataset	SLI	ENDER	$\mathbf{gHKL_{MT-\rho}}$		
			$\rho = 2$	$\rho = 1.5$	$\rho = 1.1$
BALANCE	0.758 ± 0.025 (10.4, 1.7)	0.795 ± 0.034 (112, 2.4)	$\mathbf{0.817 \pm 0.028}$ (2468.9, 2.842)	0.808 ± 0.032 (112, 1.643)	0.807 ± 0.034 (85, 1.612)
CAR	0.823 ± 0.029 (18.3, 2.93)	0.835 ± 0.024 (270, 3.05)	0.864 ± 0.02 (9571.2, 3.136)	0.86 ± 0.028 (220.3, 1.64)	$\mathbf{0.875 \pm 0.029}^*$ (269.3, 1.845)
C.M.C.	0.446 ± 0.016 (21.1, 1.9)	$\mathbf{0.485 \pm 0.015}^*$ (513, 4.36)	0.472 ± 0.014 (10299.3, 2.854)	0.463 ± 0.017 (512.9, 1.946)	0.465 ± 0.016 (396.4, 1.875)
ECOLI	0.726 ± 0.042 (7.8, 1.34)	0.636 ± 0.028 (35, 2.15)	0.779 ± 0.057 (4790.2, 2.985)	$\mathbf{0.784 \pm 0.045}^*$ (34.3, 1.05)	0.778 ± 0.054 (32.4, 1.155)
GLASS	0.43 ± 0.061 (7.4, 1.41)	0.465 ± 0.052 (70, 3.21)	0.501 ± 0.049 (5663.7, 2.396)	$\mathbf{0.525 \pm 0.043}^*$ (69.1, 1.154)	0.524 ± 0.046 (54.6, 1.038)
IRIS	0.766 ± 0.189 (2.2, 1.02)	0.835 ± 0.093 (10, 1.34)	0.913 ± 0.083 (567, 2.439)	$\mathbf{0.927 \pm 0.024}^*$ (9.8, 1)	0.893 ± 0.091 (8.6, 1)
LYMPH	0.61 ± 0.066 (2.7, 1)	0.706 ± 0.058 (34, 2.2)	0.709 ± 0.061 (4683.8, 2.299)	$\mathbf{0.724 \pm 0.078}$ (33.7, 1.008)	0.722 ± 0.078 (33, 1.01)
T.A.E.	0.334 ± 0.035 (1.1, 1)	0.41 ± 0.065 (39, 1.86)	$\mathbf{0.418 \pm 0.049}$ (5707.4, 2.253)	0.399 ± 0.049 (38.3, 1.000)	0.402 ± 0.046 (38.1, 1.052)
YEAST	0.478 ± 0.035 (23.4, 1.63)	$\mathbf{0.497 \pm 0.015}$ (218, 5.78)	0.487 ± 0.021 (8153.6, 2.847)	0.485 ± 0.022 (217.8, 1.796)	0.486 ± 0.021 (179.6, 1.726)
ZOO	0.556 ± 0.062 (7.1, 1.24)	$\mathbf{0.938 \pm 0.033}$ (33, 1.29)	0.877 ± 0.06 (3322.2, 2.695)	0.928 ± 0.037 (32.3, 1.000)	0.927 ± 0.039 (31.9, 1.014)

Table 5.4: Results on multiclass REL classification. Accuracy along with standard deviation, the number of rules and the length of rules averaged over 10 splits are shown. Statistical significance tests performed at 99% confidence.

Multiclass Classification in REL

This section summarizes the simulations with multiclass classification datasets. The details of the multiclass datasets[10] are provided in Table 5.3. The results, averaged over 10 random train-test splits with 10% train data, are presented in Table 5.4. Following [45, 46], we report the accuracy to compare generalization performance among the algorithms. The number of rules as well as the average length of the rules is also reported to give an understanding of the interpretability of the output.

Note that $\mathbf{gHKL_{MT-\rho}}$ obtains the best generalization performance in 7 datasets, out of which 4 results are statistically significant. Moreover, $\mathbf{gHKL_{MT-1.5}}$ and $\mathbf{gHKL_{MT-1.1}}$ usually

[10]Classes within a dataset with too little number of instances (< 3) are not considered for simulations. This is because we perform a 3-fold cross validation for hyper-parameter selection.

select the shortest rules among all the methods. As in the case of binary classification, the number of rules as well as the average rule length in case of $\mathbf{gHKL_{MT-2}}$ is generally very large compared to $\mathbf{gHKL_{MT-1.5}}$ and $\mathbf{gHKL_{MT-1.1}}$. This again shows the efficacy of the proposed ℓ_1/ℓ_ρ regularizer in obtaining a compact set of simple rules.

5.6 Conclusions

We generalized the HKL framework in two ways. First, a generic ℓ_1/ℓ_ρ block-norm regularizer is employed that alleviates a restriction in the sparsity pattern induced by HKL. Secondly, the framework is further generalized to handle the problem of multi-task feature learning (and termed as gHKL_{MT}). An interesting computational result is that gHKL_{MT} can be solved in time polynomial in the active set and training set sizes whenever the HKL formulation can be solved in polynomial time. The other important contribution is the application of the proposed gHKL_{MT} formulation in the learning setting of Rule Ensemble Learning (REL), where HKL has not been previously explored. Empirical results on both binary and multiclass classification for REL illustrate the efficacy of the proposed generalizations.

Chapter 6

Learning Kernels on Latent Task Structure

In the previous chapters, we considered multi-task learning (MTL) settings where all the tasks were assumed to be related. In Chapters 3 and 5, all the tasks shared the same set of selected base kernels, while in Chapter 4, they shared a common latent feature space residing within the selected base kernels. However, such an assumption may not hold in real world applications [70]. In particular, there may also exist a few *outliers tasks* — those tasks that are not related to any other task. In such cases, learning all tasks simultaneously under the assumption that all the tasks are related may lead to loss of generalization ability and worsening of performance [77, 82].

As an example, consider the problem of face recognition of different people. This can be modeled as a multi-task problem, each task being identification of a certain individual. All the tasks are related since they the set of relevant features is common for all of them. However, we also expect that depending on ethnicity, race, region, gender, *etc.*, some people will share facial characteristics more closely [121]. In genomics, organisms from same family (in terms of taxonomy) are expected to show more similar characteristics/behavior [136].

In such a scenario, learning which feature information should be shared among which tasks may lead us to better generalizability by being able learn a more accurate model for the application at hand. However, this is a challenging problem since it is not known apriori which task is related with whom under which feature space.

Contributions

We consider the multi-task setting where some relevant features (kernels) could be shared across few tasks. We propose a convex formulation that learns the feature space shared by groups of tasks that have task parameters close to each other (in the shared feature space). Since we do not have any prior information, we consider the search space consisting of all possible groups of tasks, *i.e.*, the power-set of the given tasks. The feature space within each group is learnt as a linear combination of given base kernels. In the special case where the base kernels are chosen to be linear kernels formed by individual input features, this amounts to feature selection.

The proposed formulation employs a graph-based regularizer that encodes an important structure among the groups of tasks: if there is no feature space under which a group of tasks has close-by task parameters, then there does not exists such a feature space for any of its supersets. Note that this specialty, when appropriately exploited by an algorithm, may help avoid search in potentially large portions of the search space that are anyway not fruitful. We propose an active set algorithm exploits this specialty and optimally solves the proposed formulation in a time polynomial in the number of groups discovered. Note that this number is typically very small compared to the size of the power-set of the given tasks. Experiments on benchmark datasets show that the proposed methodology achieves good generalization and outperforms state-of-the-art multi-task learning techniques in some cases.

Outline

The rest of the chapter is organized as follows. Section 6.2 formalizes the notation and the problem setup. The details of the proposed formulation and the algorithm for solving it are discussed in Sections 6.3 and 6.4 respectively. Experimental results are discussed in Section 6.5. Section 6.6 summarizes the work and the key contributions. We begin by discussion the related works in the next section.

6.1 Related Work

Recent works in multi-task literature have focused on the problem of learning groups of tasks that are more related than others. In a hierarchical Bayesian setup, [139] employed the Dirichlet process to learn groups of tasks having close-by task parameters. On the other hand, [68] proposed a k-means clustering based algorithm to cluster such tasks. Note that these approaches did not incorporate feature learning within groups. Another work, [77], grouped the task sharing a low dimensional subspace [8]. [68, 77] propose non-convex formulations in empirical risk regularized framework (2.1) and require the number of clusters as a pre-defined parameter. They learn disjoint groups of tasks, *i.e.*, a task belongs to only one group.

Note that the disjoint groups setting may not always reflect the true task structure [82, 151]. Hence, some works aim at discovering groups of tasks that may have overlap between them, *i.e.* a task may belong to more than one group. [82] assume the tasks to be a sparse combination of underlying meta-tasks. Tasks sharing meta-tasks are grouped together. [151] proposes to decompose the task parameters as a superposition of task specific and cluster specific parameters. Two tasks belong to a group if they have identical cluster specific parameter for the input feature. Another work, [136], attempts to search for suitable groups among all possible groups of tasks. However the run-time of their algorithm is exponential in the number of tasks.

6.2 Problem Setup

Consider a set $\mathcal{T}$ of learning tasks, T in number. The training data for the t^{th} task is denoted by: $\mathcal{D}_t = \{(\mathbf{x}_{ti}, y_{ti}),\ i = 1, \ldots, m\}\ \forall t = 1, \ldots, T$, where $(\mathbf{x}_{ti}, y_{ti})$ represents the i^{th} input/output pair of the t^{th} task. For the sake of notational simplicity, we assume that the number of training examples is the same for all the tasks. The task predictors are assumed to be affine: $F_t(\mathbf{x}) = \langle f_t, \phi(\mathbf{x})\rangle - b_t,\ t = 1, \ldots, T$, where f_t is the task parameter of the t^{th} task, $\phi(\cdot)$ is the feature map and b_t is the bias term.

Recall that our aim is to learn kernels shared by groups of related tasks from the power-set of $\mathcal{T}$ (henceforth denoted by $\mathcal{V}$). To this end, we propose an additive model for task parameters, $f_t = \sum_{w \in \mathcal{G}_t} f_{tw}$ where $\mathcal{G}_t$ is the set of all subsets of $\mathcal{T}$ containing task t. The parameter f_{tw} indicate the influence of feature space induced by the shared kernel within the group/subset w on task t. As discussed previously, the shared kernel for each group of tasks is learnt as a conic combination of given base kernels. To this end, let $k^1, \ldots, k^n$ be the given base kernels. Let $\phi^j(\cdot)$ denote the feature map induced by the j^{th} kernel k^j, $j = 1, \ldots, n$. Hence, $\phi(\mathbf{x}) =$

$(\phi^1(\mathbf{x}), \ldots, \phi^n(\mathbf{x}))$. Let f_{tw}^j represent the projection of f_{tw} onto the ϕ^j space. In other words, $f_{tw} = (f_{tw}^1, \ldots, f_{tw}^n)$. With this notation, the prediction function for the task t can be rewritten as $F_t(\mathbf{x}) = \langle f_t, \phi(\mathbf{x})\rangle - b_t = \sum_{w\in\mathcal{G}_t}\langle f_{tw}, \phi(\mathbf{x})\rangle - b_t = \sum_{w\in\mathcal{G}_t}\sum_{j=1}^{n}\langle f_{tw}^j, \phi^j(\mathbf{x})\rangle - b_t$. Note that if $f_{tw}^j = 0 \;\forall\; t \in w$, then the feature space corresponding to the j^{th} kernel is absent in the feature space corresponding to the shared kernel of the group w. Hence learning the the task predictors or equivalently the task parameters f_{tw}^j and the bias terms b_t amounts to learning which kernels (feature spaces) is shared among which tasks. In the subsequent section, a novel convex formulation for learning the optimal task predictors in the current setup is presented.

6.3 Graph-based Multi-task Regularization

In this section, we present the proposed convex feature learning formulation for learning kernels on the latent task structure. As stated previously in Chapter 2.1, we employ the regularized risk minimization framework [125] and consider the following problem:

$$\min_{f_t, b_t \forall t\in\mathcal{T}} \Omega(f_1, \ldots, f_T)^2 + C\sum_{t=1}^{T}\sum_{i=1}^{m} V(F_t(\mathbf{x}_{ti}), y_{ti}) \tag{6.1}$$

where $\Omega(f_1, \ldots, f_T)^2$ is the regularizer, $V(\cdot,\cdot)$ is a suitable convex loss function (like the hinge loss) and C is the regularization parameter. The regularizer $\Omega(\cdot)$ is usually chosen based on some prior knowledge on the relationships among the tasks. For instance, in cases it is known that all the task parameters are close to each other, the following regularizer may be employed [48]:

$$\Omega(f_1, \ldots, f_T)^2 = \mu\|h_0\|_2^2 + \sum_{t=1}^{T}\|h_t\|_2^2, \tag{6.2}$$

where $f_t = h_0 + h_t \;\forall\; t = 1, \ldots, T$ and $\mu > 0$ is a given parameter that controls the trade-off between regularizing the mean (h_0) and the variance $(h_1, \ldots, h_T)$ parameters of all the tasks.

In the following text, we present a novel regularizer suitable for the current problem. We begin by writing down a basic term in the proposed regularizer Θ_w^j, which promotes proximity among task parameters within the group w with respect to the feature space j (induced by k^j):

$$\Theta_w^j = \left(\mu\|h_{0w}^j\|_2^2 + \sum_{t\in T_w}\|h_{tw}^j\|_2^2\right)^{\frac{1}{2}}$$

where $f^j_{tw} = h^j_{0w} + h^j_{tw}$. This term is motivated from (6.2). Note that $(\Theta^j_w = 0) \Rightarrow f^j_{tw} = \mathbf{0}\ \forall\, t \in w$. Hence, if Θ^j_w is zero, then the tasks within the group w do not share[1] the kernel k^j.

The terms $\Theta^j_w,\ j = 1, \ldots, n$, are now combined using a p-norm, leading to: $\|\Theta_w\|_p = \left(\sum_j \left(\Theta^j_w\right)^p\right)^{\frac{1}{p}}$ where Θ_w is the vector with entries as $\Theta^j_w,\ j = 1, \ldots, n$, and $p \in (1, 2)$. The p-norm promotes sparse kernel combination for the group w (*i.e.*, forces many $\Theta^j_w = 0$). With the interpretation of Θ^j_w noted above, this essentially enables kernel (feature) learning within the group of tasks w. In addition, $(\|\Theta_w\|_p = 0) \Rightarrow (\Theta^j_w = 0\ \forall\, j = 1, \ldots, n) \Rightarrow (f^j_{tw} = \mathbf{0}\ \forall\, t \in w,\ j = 1, \ldots, n)$. Hence, in case $\|\Theta_w\|_p = 0$, then the node w does not contain related tasks since the tasks within the group w do not share any base kernels and the group does not influence any of the task prediction functions F_t.

With this interpretation, one naive way of obtaining few promising groups of related tasks that share a feature space is by employing a ℓ_q-norm, $q \in (1, 2)$, over the terms $\|\Theta_w\|_p$, $w \in \mathcal{V}$: $\Omega(f_1, \ldots, f_T) = \left(\sum_{w \in \mathcal{V}} \left(\|\Theta_w\|_p\right)^q\right)^{\frac{1}{q}}$. However, the problem with this regularizer is that it renders the formulation (6.1) infeasible for real-world applications as the resultant optimization problem cannot be, in general, solved in a time polynomial in the number of tasks ($|\mathcal{V}| = 2^T - 1$).

One key idea is to employ a graph-based regularizer that alleviates this problem by exploiting a special structure among the groups of tasks. Note that the groups of tasks can be represented as nodes of a directed acyclic graph with the partial order $\subseteq$, representing the "subset of" relation. It can verified that $\langle \mathcal{V}, \subseteq \rangle$ is a lattice. The topmost node of the lattice represents a *dummy* node – the group with no tasks in it, the second level nodes represents groups with single task and so on. The bottommost node represents the group consisting of all the T tasks. As discussed earlier, we would like to encode into our regularizer the following structure among the groups of tasks: if there is no feature space under which a group of tasks has similar task parameters, then there does not exists such a feature space for any of its supersets. In the context of the present lattice, this is same as saying: if a node w is not selected, then the entire sub-lattice $D(w)$, which consists of all the descendants of w (including w itself), need not be selected. In the following, a regularizer that reflects this special structure is presented.

Motivated by the graph-based regularizers employed in [148, 11] as well as the one dis-

[1]In case of positive definite base kernels, it can also be shown that if Θ^j_w is non-zero, then *all* the tasks within the group w share the kernel k^j, *i.e.*, $(\Theta^j_w \neq 0) \Rightarrow f^j_{tw} \neq \mathbf{0}\ \forall\, t \in w$.

cussed in Chapter 5, we propose the following novel regularizer for the problem at hand:

$$\Omega(f_1, \dots, f_T) = \sum_{v \in \mathcal{V}} d_v \left(\sum_{w \in D(v)} \|\Theta_w\|_p^q \right)^{\frac{1}{q}} \tag{6.3}$$

where $q \in (1, 2)$, $p \in (1, 2)$ and d_v is a parameter that enables encoding prior knowledge regarding the task-relatedness in the group/node v. Note that the proposed regularizer (6.3) may also be viewed as a $\ell_1/\ell_q/\ell_p$ mixed-norm regularizer. The ℓ_1-norm over the nodes ($v \in \mathcal{V}$) of the lattice promotes sparsity, and hence we have $\left(\sum_{w \in D(v)} \|\Theta_w\|_p^q\right)^{\frac{1}{q}} = 0$ for most $v \in \mathcal{V}$, *i.e.*, few groups of related tasks are selected. Moreover, $\left(\sum_{w \in D(v)} \|\Theta_w\|_p^q\right)^{\frac{1}{q}} = 0 \Rightarrow f_{tw}^j = 0\, \forall\, t \in w,\ \forall\, j \in 1, \dots, n,\ \forall\, w \in D(v)$. In other words if a group is not selected by the 1-norm, then none of its descendants are selected by the formulation — the special structure we wanted to encode. The q-norm brings in additional sparsity among the descendants of the groups that are selected by the 1-norm. As we detail later, the key advantage with this regularizer is that it renders the proposed formulation (6.1), solvable in reasonable time.

In the following, a specialized dual of (6.1) with the proposed regularizer (6.3) is presented. This gives further insights into the working of the proposed formulation and motivates an efficient active set algorithm for solving it. In order to keep notations simple, the dual is presented for the case where each of the given tasks is a binary classification problem and the loss function $V(F_t(\mathbf{x}), y)$ is the hinge loss: $\max(0, 1 - yF_t(\mathbf{x}))$. However, it is easy to extend the derivations to other learning settings and convex loss functions as well.

Theorem 6.3.1. *In the case where the given tasks are all of binary classification and the hinge loss is employed as the loss function, the dual of (6.1) with the regularizer defined in (6.3) is given by*

$$\min_{\eta \in \Delta_{|\mathcal{V}|}} \quad H(\eta) \tag{6.4}$$

where $\mathbf{z} \in \Delta_{s,r} \equiv \{\mathbf{z} \in \mathbb{R}^s \mid \mathbf{z} \geq 0, \sum_{i=1}^s \mathbf{z}_i^r \leq 1\}$ *and* H *is a convex function with* $H(\eta)$ *equal to the optimal value of the following optimization problem:*

$$\max_{\alpha_t \in \mathbb{R}^m \forall t} \sum_t \mathbf{1}^\top \alpha_t - \frac{1}{2} \left(\sum_{w \in \mathcal{V}} \zeta_w(\eta) \left(\sum_{j=1}^n \left(\alpha^\top \mathbf{K}_w^j \alpha\right)^{\bar{p}} \right)^{\frac{\bar{q}}{\bar{p}}} \right)^{\frac{1}{\bar{q}}} \tag{6.5}$$

$$\text{s.t. } \mathbf{0} \leq \alpha_t \leq C\mathbf{1}\, \forall\, t, y_t^\top \alpha_t = 0\, \forall\, t,$$

where y_t denotes the vector with entries as y_{ti}, $\alpha = [\alpha_1 \ \ldots \ \alpha_T]^\top$, $\mathbf{1}$ and $\mathbf{0}$ denote vectors with all entries as 1 and 0 respectively, $\zeta_w(\eta) = \left(\sum_{v \in A(w)} d_v^q \eta_v^{1-q}\right)^{\frac{1}{1-q}}$, $A(w)$ represents the set of ancestors for node w (including w), $\bar{p} = \frac{p}{2(p-1)}$, $\bar{q} = \frac{q}{2(q-1)}$. The easiest way to describe the matrix $\mathbf{K}_w^j \in \mathbb{R}^{mT \times mT}$ is by writing it as a block matrix of size $T \times T$ with the $(t_1, t_2)^{th}$ block as the matrix $\mathbf{K}_w^j(t_1, t_2) \in \mathbb{R}^{m \times m}$. The $(i_1, i_2)^{th}$ entry of $\mathbf{K}_w^j(t_1, t_2)$ is:

$$(\mathbf{K}_w^j(t_1, t_2))_{i,j} = \begin{cases} \frac{\mu+1}{\mu} y_{t_1 i_1} y_{t_2 i_2} k^j(\mathbf{x}_{t_1 i_1}, \mathbf{x}_{t_2 i_2}) & \text{if } t_1 = t_2 \in \ w, \\ \frac{1}{\mu} y_{t_1 i_1} y_{t_2 i_2} k^j(\mathbf{x}_{t_1 i_1}, \mathbf{x}_{t_2 i_2}) & \text{if } t_1, t_2 \in w, \ t_1 \neq t_2, \\ 0 & \text{otherwise.} \end{cases}$$

Proof is provided in appendix A.2.1.

The dual (6.4) provides interesting insights into the formulation. To this end, let us begin with an interpretation for the $\mathbf{K}_w^j$ matrices. From their definition, it is easy to see that $\mathbf{K}_w^j$ can also be viewed as the gram matrix of training examples from all the tasks with an appropriately defined kernel function k_w^j. As $\mu \to 0$, $\frac{1}{\mu}$ dominates and the kernel function k_w^j reflects great similarity between examples of tasks in w (and vice-versa). Also, the examples from tasks not belonging to w have low similarity with those in w. Hence, the kernel k_w^j captures the similarity between the tasks in the group w under the j^{th} feature space.

Now lets focus on the problem (6.5). In the special case $\bar{p} = \bar{q}$, this problem is same as the $\ell_{\hat{q}}$-MKL formulation [80] with $\hat{q} = \frac{\bar{q}}{\bar{q}-1}$ and with base kernels as $\hat{k}_w^j = (\zeta_w(\eta))^{\frac{1}{\bar{q}}} k_w^j \ \forall w \in \mathcal{V}$ and $\forall j = 1, \ldots, n$. Hence the problem (6.5) realizes a sparse combination of these kernels. With the interpretation provided above, this essentially amounts to a sparse selection of kernels shared within groups of tasks. However, unlike $\ell_{\hat{q}}$-MKL that performs a 'flat' kernel selection, the kernels are weighted by a function $\zeta_w(\eta)$ that induces structured sparsity.

To see this, let us shift our focus to the dual problem (6.4). Because of the simplex constraint over η (the ℓ_1-norm regularization), most of the η_v will be zero at optimality. From the definition of $\zeta_w(\eta)$, we obtain: $(\eta_v = 0) \Rightarrow \zeta_w(\eta) = 0 \ \forall \ w \in D(v)$. In addition, $\zeta_w(\eta) = 0$ implies the entire set of kernels $k_w^1, \ldots, k_w^n$ are not selected. Hence, $\zeta_w(\eta) = 0$ implies that the group of tasks in w do not share any feature space and group w is not selected. To summarize, few groups of tasks that share a feature space are selected and if a group of tasks is not selected $(\eta_v = 0)$, then all its supersets are not selected (as $\zeta_w(\eta) = 0 \ \forall \ w \in D(v)$).

Figure 6.1 provides an illustration of the dual problem for the case of four tasks ($T =$

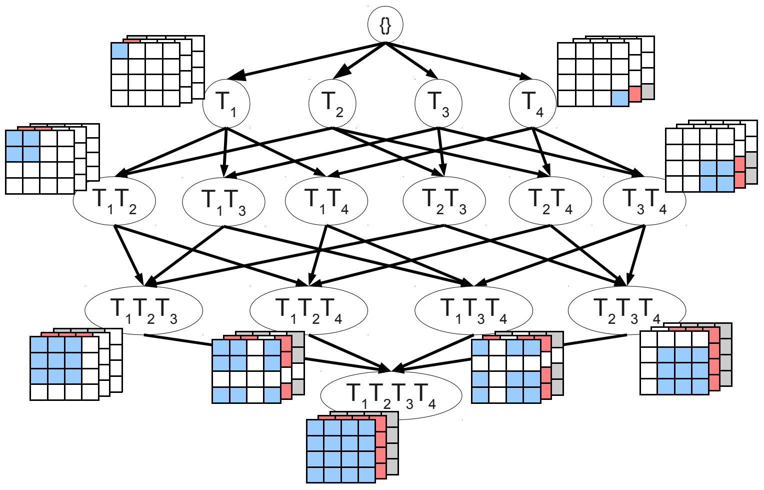

Figure 6.1: Figure illustrating the arrangement of kernels (k_w^j) selected by the dual (6.4). Four tasks, $\mathcal{T} = \{T_1, T_2, T_3, T_4\}$, and three base kernels (k^1, k^2, k^3) are considered. The base kernels are represented by three different colors (blue, red and grey). The kernel matrices corresponding to several groups of tasks are shown. Note that these matrices have a block structure with zeros for the entries corresponding to the tasks absent in the group. Colored portions in a matrix indicate the entries corresponding to the tasks present in the group. Figure best viewed in color.

4) and three base kernels ($n = 3$). The proposed formulation is equivalent to performing a structured selection among $n \times |\mathcal{V}|$ kernels arranged on the lattice. At each node, there are n kernels that represent the relatedness of tasks in that group. The subsequent section, presents an efficient active set algorithm for solving the proposed formulation that exploits the special structure in the solution described above.

6.4 Optimization Algorithm

Following a common practice for solving large-scale convex sparsity problems [85, 11], we propose solving the dual (6.4) using an active set algorithm. We had employed it in Chapter 5. For the sake of completeness, we begin by having a brief discussion of it, in the context of the present problem.

The basic idea of the active set algorithm is as follows: the formulation is solved iteratively using improved guesses for the active set, which is defined as the set of w for which $\eta_w \neq 0$ at optimality. At each iteration the problem restricted to the variables in the active set is solved using an appropriate solver. In order to save computational cost, the size of the initial active set is usually taken to be minimal. After solving the problem with the variables restricted to the current active set, a sufficiency condition for optimality of the solution is verified. In case the solution is optimal, the algorithm terminates. In case it is not, the active set is updated and the restricted problem with the new active set is solved. This process is repeated until optimality is reached. Any prior knowledge related to the structure of the optimal solution may also be incorporated in building the active set at each iteration. In case of problems like (6.4) with sparse solutions, the hope is that one may not solve the problem with all the variables, η_w, which are exponential in T in number.

In order to formalize the active-set algorithm, we need i) an initial guess for the active set and a procedure for building/improving the active set after each iteration, ii) a sufficiency condition for verifying optimality of the current solution. Ideally, the complexity of verification of the condition should depend on the active set size rather than the problem size iii) an efficient algorithm for solving the formulation restricted to active set.

We begin with the first. Let $\mathcal{W}$ represent the active set. The initial guess as well as the methodology for the identification of the promising nodes is motivated by the special structure in the solution of (6.4): if a node w is not selected in the optimal solution, *i.e.*, $\eta_w = 0$, none of the descendants of w are selected (since $\zeta_v(\eta) = 0 \ \forall v \in D(w)$). Equivalently, it can be stated that a node w can be selected only if *all* its ancestors are selected, *i.e.*, only if $\eta_v \neq 0 \ \forall v \in A(w)$. Due to this observation, $\mathcal{W}$ is always maintained to be equal to its hull, where $hull(\mathcal{W})$ is defined to be the set of all the ancestors of the nodes in $\mathcal{W}$. Accordingly, the initial guess for $\mathcal{W}$ is taken to be the second level nodes, *i.e.*, the singleton task groups. Also, in the subsequent iterations, only those nodes which have all their parents in $\mathcal{W}$ are considered as potential candidates for entry inside $\mathcal{W}$. Towards the second requirement, we present the following theorem

Theorem 6.4.1. *For a given active set $\mathcal{W}$ such that $\mathcal{W} = hull(\mathcal{W})$, let the optimal solution of*

(6.4) restricted to $\mathcal{W}$ be $(\hat{\eta}, \hat{\alpha})$. Define $\hat{\Theta}$ to be as follows:

$$\hat{\Theta} = \left(\sum_{w \in \mathcal{V}} \hat{\zeta}_w(\hat{\eta}) \left(\sum_{j=1}^{n} \left(\hat{\alpha}^\top \mathbf{K}_w^j \hat{\alpha} \right)^{\bar{p}} \right)^{\frac{\bar{q}}{\bar{p}}} \right)^{\frac{1}{\bar{q}}}$$

Then, $(\hat{\eta}, \hat{\alpha})$ is an optimal solution of (6.4) with duality gap ϵ if:

$$\max_{s \in sources(\mathcal{W}^c)} \left(\sum_{w \in D(s)} \frac{\hat{\alpha}^\top \left(\sum_{j=1}^{n} \mathbf{K}_w^j \right) \hat{\alpha}}{\left(\sum_{v \in A(w) \cap D(s)} d_v \right)^2} \right) \leq \hat{\Theta} + 2\epsilon \tag{6.6}$$

where $sources(\mathcal{W})$ is the set of nodes in $\mathcal{W}$ with no parent in $\mathcal{W}$ and $\mathcal{W}^c$ denotes the set of all the nodes present in the lattice $\mathcal{V}$ but not in $\mathcal{W}$.

Proof is provided in appendix A.2.2.

As mentioned earlier, the above sufficiency condition is useful only if it can be verified in polynomial time in $|\mathcal{W}|$. Firstly, size of $sources(W^c)$ is upper-bounded by $T|\mathcal{W}|$. The denominator in the summation, $\sum_{v \in A(w) \cap D(s)} d_v$ can be computed in $O(T)$ provided d_v is decomposable as a product. In the simulations we use, $d_v = 1.5^{|v|}$. Because of the block structure of the matrices $\mathbf{K}_w^j$, the sum over descendants in (6.6) can be computed in $O(T^2 m^2)$.

In the following, we present an efficient algorithm of solving (6.4) restricted to $\mathcal{W}$. Note that (6.4) has a simple constraint set, which is a simplex and the gradient $\nabla H(\eta)$ can be computed using the Danskin's theorem [23]. The i^{th} component of this gradient is given by

$$(\nabla H(\eta))_i = -\frac{d_i^q \eta_i^{-q}}{2\bar{q}} \left(\sum_{w \in \mathcal{V}} \zeta_w(\eta) \left(\sum_{j=1}^{k} \left(\bar{\alpha}^\top \mathbf{K}_w^j \bar{\alpha} \right)^{\bar{p}} \right)^{\frac{\bar{q}}{\bar{p}}} \right)^{\frac{1}{\bar{q}}-1} \left(\sum_{w \in D(i)} \zeta_w(\eta)^q \left(\sum_{j=1}^{k} \left(\bar{\alpha}^\top \mathbf{K}_w^j \bar{\alpha} \right)^{\bar{p}} \right)^{\frac{\bar{q}}{\bar{p}}} \right)$$

where $\bar{\alpha}$ is an optimal solution of (6.5) with the given η. Hence one can employ projected gradient-descent algorithm or any of its variants for solving (6.4. Here we employ the mirror-descent algorithm [22] for solving (6.4). Note that the gradient computation $\nabla H(\eta)$ at $\eta = \eta'$ requires solving (6.5) at η'. Since the constraint set in (6.5) is similar to that in an SVM, we employ sequential minimal optimization (SMO) algorithm [105] for solving (6.5). Algorithm 4 summarizes the proposed active-set method.

The computational complexity of the active set algorithm is as follows: let the final size of the active set be W. Hence, (6.4) is solved a maximum of W times. Each run of mirror-descent

Algorithm 4 Active Set Algorithm

```
**Input:** Training data $\mathcal{D}_t \; \forall t$, tolerance $\epsilon$
**Output:** $\eta, \alpha, \mathcal{W}$
Initialize $\mathcal{W} = \{w | \; w \in \mathcal{V}, \; |T_w| = 1\}$ (first level nodes of the lattice $\mathcal{V}$)
**repeat**
    For the current $\mathcal{W}$, solve for $\eta, \alpha$ in (6.4) using mirror-descent & SMO
    Calculate $V =$ nodes violating the condition (6.6)
    Update $\mathcal{W} = \mathcal{W} \cup V$
**until** V is empty
```

algorithm takes $O(log(W))$ iterations [22] while in each iteration the dominant computation is that of SMO for solving (6.5). A conservative complexity estimate for the SMO algorithm is $O((Tm)^3(Wn)^2)$. The computing cost for kernel matrices is $O(n(Tm)^2)$, while that of verifying the sufficiency condition is $O((Tm)^2TW)$. Thus, the overall complexity is $O(n^2W^2m^3T^3)$.

We end this section by presenting a variant of the proposed methodology. The motivation for this variant comes from a closer-look at the complexity of the active-set algorithm. Since the active-set always satisfies the condition $\mathcal{W} = hull(\mathcal{W})$ and since the complexity depends on the active-set size W; in practice one cannot realize situations where the group selected is way down the lattice. In other words, it is rare that a group with large number of tasks is selected. However, as shown in simulations, there may exist applications where the task parameters are extremely close-by for all or most of the tasks. Hence realizing a group containing most of the tasks may be beneficial. One simple modification of the proposed methodology for selecting such large groups is: invert the lattice of groups of tasks, *i.e.*, revert the parent-child relations, and employ exactly the same formulation (the descendants become the ancestors and vice-versa). It is easy to see that in this case groups involving large number of tasks may be selected; whereas selecting groups involving few tasks is now improbable. Though this modification is simple and interesting, the natural motivation for employing the graph-based regularizer is absent in this case. The graph-based regularizer needs to be motivated purely from a computational perspective in this case.

6.5 Experiments

In this section we present our empirical studies on the following benchmark multi-task classification and regression datasets.

Sarcos A multiple-output regression dataset used in [146]. The aim is to predict inverse dynamics corresponding to the seven degrees-of-freedom of a robot arm. The number of tasks is 7 and there are 21 real valued input features. Following [146], we sampled 2000 random examples from each task.

Parkinson A multi-task regression dataset from the UCI repository [50]. The aim is to predict Parkinson's disease symptom score for patients at different times using 19 bio-medical features. The dataset has 5,875 observations for 42 patients. The symptom score prediction problem for each patient is considered as a regression task. Thus, there are 42 regression tasks[2] with number of instances for each task ranging from 101 to 168.

Yale A face recognition dataset from Yale face[3] base. It contains 165 images of 15 subjects. Following the experimental setup in [145], each task is defined as the binary classification problem of classifying two subjects. Thus there are 28 tasks and the number of features is 30.

Landmine A benchmark multi-task classification dataset used in [139, 145]. It contains examples collected from various landmine fields. Each example is represented as a 9-dimensional real valued feature vector. Each task is a binary classification problem with the goal being to predict landmines (positive class) or clutter (negative class). Following [139, 145], we jointly learn 19 tasks from the landmine fields numbered $1 - 10$ and $16 - 24$ in the data set. Number of instances in each task varies from 445 to 690. The dataset is highly biased against the positive class.

MHC-I A multi-task classification dataset used in [68]. It contains binding affinities of various peptides with different MHC-I molecules. Each task here is a binary classification

[2]Note that this setting is different that the one employed in the experiments discussed in Chapter 4.

[3]Dataset and train-test splits available at `http://www.zjucadcg.cn/dengcai/Data/FaceData.html`. We used the first 10 splits containing 5 training examples per subject.

problem. We perform experiments on the same 10 tasks reported in [68]. Total number of instances in the 10 tasks is 1200 and the input space consists of 180 binary features. The number of instances per task varies from 59 to 197 and the the dataset is biased against the positive class.

Letter A multi-task classification dataset used in [75]. It consists of handwritten letters from different writers. Each task is a binary classification problem of distinguishing between pairs of letters. There are 9 such binary classification tasks and we randomly sampled 300 data points per task for our simulations. The input features used are the $8 \times 16 = 128$ binary pixels.

Each dataset was further randomly split into training and test sets. In Landmine, MHC-I and Letter datasets, random 20%-80% train-test splits were considered. In the case of Sarcos dataset 15 random samples per task were used for training and the rest for testing. For Parkinson dataset, 5 random examples per task were used in training and the rest for testing. We compare the generalization performace of the following algorithms.

STL A baseline approach in which the tasks are learned independently using SVM [125].

MK-MTFL The multiple kernel multi-task feature learning formulation (3.5) proposed in Chapter 3. As in previous experiments, the parameters (p, q) were fixed to $(6, 1)$, thereby promoting sparsity in kernel selection and allowing the learnt feature space to be a function of both shared as well as task-specific parameters.

MTL A classical multi-task learning algorithm [48]. It assumes that all tasks are related and have the task parameters in proximity. It does not incorporate feature learning.

CMTL The clustered multi-task learning formulation proposed in [68]. It finds clusters of tasks having task parameters in proximity. It does not incorporate feature learning. Its code is available at `http://cbio.ensmp.fr/~ljacob/documents/cmtl-code.tgz`.

DMTL The multi-task feature learning formulation in [70]. It performs feature selection to discover features shared across all the tasks as well as task-specific features. It also induces proximity among the task parameters in the shared feature space. Its code is available at `http://www.ali-jalali.com/index_files/L1Linf_LASSO.r`.

MTGFL The proposed multi-task group and feature learning formulation. The base kernels were taken as linear kernels with individual input features. In addition, the linear kernel using all input features was also employed as a base kernel. Thus if the input space dimensionality is n, then we generate $n + 1$ linear kernels. We did not employ non-linear kernels in order to be fair and comparable with CMTL and DMTL. The parameters p, q were both fixed at 1.5, promoting sparsity in selecting groups of related tasks as well as in selecting the kernel induced feature spaces. Since the base kernels include individual input-feature based linear kernels, this amounts to feature selection. Its code is available at `www.cse.iitb.ac.in/~pratik.j/MTFL_icml12.tar.gz`.

Note that all of the above multi-task learning techniques rely on the same notion of task-relatedness: task parameters of related tasks are close. Hence a comparison among them is indeed meaningful.

The free parameters in all the methods were tuned using nested 3-fold cross validation procedure. The details of the parameter ranges are as follows: for **MTGFL**, **MTL**, **MK-MTFL** and **STL**, the regularization parameter C was chosen from the set $\{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$. **MTGFL** and **MTL** have an additional parameter μ, which was chosen from the set $\{10^{-3}, 10^{-2}, \ldots, 10\}$. **CMTL** has 4 parameters and we considered 3 values for each leading to $3^4 = 81$ combinations [145]. **DMTL** has 2 parameters and we considered 7 values of each leading to 49 combinations.

In case of regression datasets we report the explained variance, whereas for classification datasets we report AUC (Area Under Curve). In both cases, higher the value reported, the better the algorithm. Also, we report both the mean and standard deviation in the values over 10 random train-test splits. The best result in each dataset is highlighted. In case the best result is with the proposed method (**MTGFL**) and its improvement over state-of-the-art is statistically significant, then we additionally mark it with a '*'. In case the best result is with an existing method and its improvement over the proposed method (**MTGFL**) is statistically significant, then we again mark it with a '*'. Statistical significance test is performed using the paired t-test

Dataset	STL	MK-MTFL	MTL	CMTL	DMTL	MTGFL
			Regression Datasets – Explained Variance (%)			
SARCOS	40.47 ± 7.56	14.23 ± 18.98	34.50 ± 10.19	33.02 ± 13.42	40.59 ± 10.24	$\mathbf{49.86 \pm 6.34}^*$
PARKINSON	2.84 ± 7.51	8.94 ± 10.17	4.94 ± 19.95	2.74 ± 3.62	-12.25 ± 7.41	$\mathbf{16.79 \pm 10.81}^*$
			Classification Datasets – AUC (%)			
YALE	93.36 ± 2.33	92.78 ± 2.79	96.35 ± 1.64	95.20 ± 2.12	92.44 ± 2.81	$\mathbf{96.98 \pm 1.55}^*$
LANDMINE	74.60 ± 1.55	75.74 ± 1.23	76.42 ± 0.78	75.86 ± 0.66	65.86 ± 2.63	$\mathbf{76.44 \pm 0.92}$
MHC-I	69.25 ± 2.07	64.51 ± 2.83	72.28 ± 1.94	$\mathbf{72.56 \pm 1.36}^*$	58.39 ± 5.46	71.68 ± 2.20
LETTER	$\mathbf{61.24 \pm 0.82}$	60.15 ± 0.61	61.02 ± 1.56	60.52 ± 1.09	59.34 ± 1.39	60.45 ± 1.75

Table 6.1: Performance of various methods on regression and classification datasets, measured in terms of percentage explained variance and area under the curve respectively. Mean and standard deviations are shown as well as the average time in minutes taken by each method within parentheses. **MTGFL** achieves better generalization as compared to state-of-the-art **CMTL** and **DMTL** formulations.

at 90% confidence.

Results of the simulations are summarized in Table 6.1. The proposed method outperformed state-of-the-art in both the regression datasets and achieved significant improvement in case of the YALE dataset. Note that in case of SARCOS dataset, the baseline **STL** performs better than **MTL** showing that there may be some tasks that are not related to some others and the latent task structure is non-trivial. Hence, learning which kernels are shared among which tasks is indeed important in this case. The excellent performance of the proposed method on these datasets indicates that it learns relevant kernel combinations on the latent task structure.

According to the results, the proposed methodology does not seem to improve over state-of-the-art in case of the MHC-I and LETTER datasets. A closer look at the datasets and the predictors achieved with state-of-the-art showed that the task parameters of *all* the tasks are quite close to each other in these datasets. This motivated us to try the inverted lattice trick described towards the end of Section 6.4. With this modified methodology we achieved an improved average AUC of 72.77% and 61.12% respectively on MHC-I and Letter datasets.

Figure 6.2 compares the learning rates of the proposed multi-task approach and the single task learning baseline as a function of training examples in SARCOS and MHC-I datasets. We

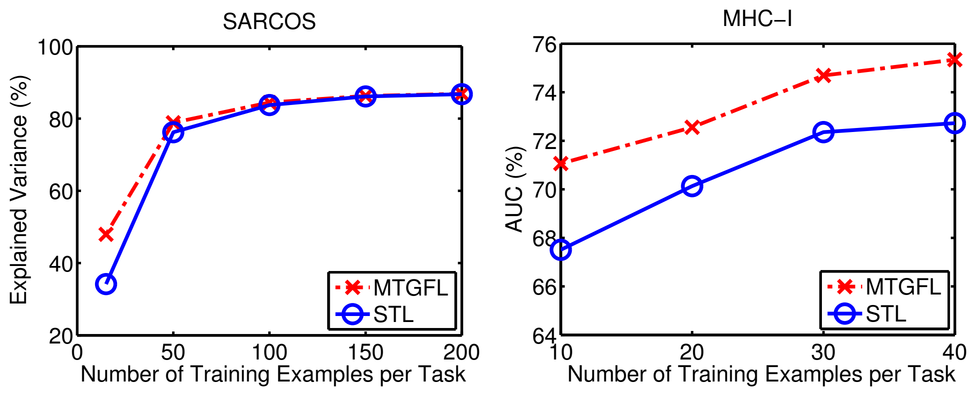

Figure 6.2: Generalization obtained by multi-task and single task learning methods on the test set as the number of training examples per task is varied. The advantage of the proposed multi-task approach **MTGFL** over **STL** is more pronounced at low training data region. In **Sarcos** dataset, **STL** starts giving almost the same performance as **MTGFL** when the number of training examples per task become 100. However, in **MHC-I** dataset, **MTGFL** is able to maintain a considerable improvement over **STL** even at high training data region.

can observe that the advantage of the proposed **MTGFL** over the single task learning baselines is high at low training data region. This advantage starts decreasing as the number of training examples increase. This is expected because with more training data, a task should be able to obtain good generalization ability on its own.

We end this section with a discussion on the run-time of the proposed method. Note that none of the existing methods attempt an optimal search over the exponentially large space of groups of tasks. Hence, as expected, the run-time of the proposed algorithm is on the higher side. On a Xeon machine with 16GB RAM, with tuned parameters, the run-time of **STL**, **MK-MTFL**, **MTL** and **CMTL** on all the datasets was less than one minute. **DMTL** also took less than one minute on all datasets except Yale and MHC-I, where it took 9 minutes and 4 minutes respectively. The run-time in minutes of **MTGFL** on various datasets are as follows: Sarcos (2), Parkinson (23), Yale (18), Landmine (14), MHC-I (15) and Letter (12). It is interesting to note that the extremely large search space (2^{42}) in case of the Parkinson dataset is searched in a reasonable time of 23 minutes. Moreover, in most datasets the proposed method achieves better generalization.

6.6 Conclusions

In real-world applications, it is important to learn which kernels (features) are shared among which tasks. This is because all the tasks may not be related to each other. We proposed a convex formulation for such a setting. The proposed graph based multi-task regularizer leverages kernel (feature) overlap among the tasks, wherever it exists, without enforcing additional restrictions over the other tasks. The regularizer encodes an important property that helps us to perform a clever search over the exponentially large space of groups of tasks. Experimental results illustrate the efficacy of the proposed approach.

Chapter 7

Kernel Selection Path in ℓ_p-norm Multiple Kernel Learning

In the previous chapters, we proposed various kernel learning formulations for multiple tasks. A common theme in the proposed formulations was to learn the kernel shared by multiple tasks as a combination of given base kernels — a setting popularly known as Multiple Kernel Learning (MKL) [83]. As discussed in Chapter 2.2, the coefficients of this combination may be regularized by a p-norm, $p \in [1, 2)$, to learn sparse combination of base kernels. In general, lower value of p promotes sparser kernel combination. The value of p is usually set by the user depending on the application at hand. However, the most suitable value of p is heavily dependent on the available data. As can be observed from the empirical results in the previous chapters and in existing p-norm MKL (or ℓ_p-MKL) works [128, 104, 81], the generalization accuracy and the sparsity of the model vary significantly with p.

Existing state-of-the-art ℓ_p-MKL algorithms are computationally too expensive to be involved thousands of times to determine the entire kernel selection path, *i.e.*, the variation in the classification accuracy as the fraction of selected kernels is varied from null to unity. Hence, there is a need to develop formulations and algorithms for efficiently computing the kernel selection path. In this chapter, we present our investigation on p-norm path following in Multiple Kernel Learning.

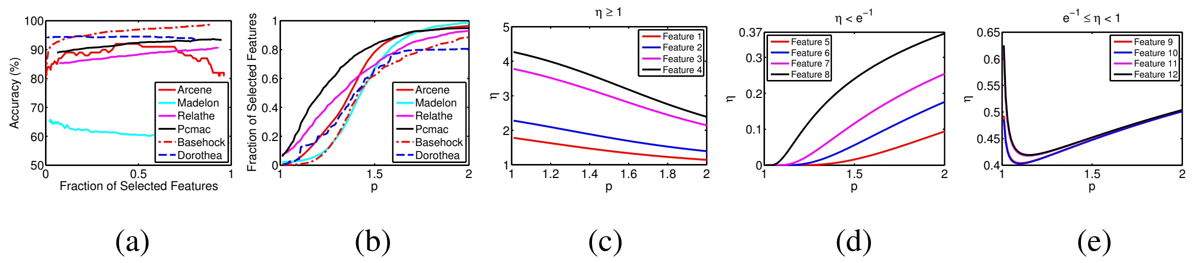

Figure 7.1: (a) The kernel selection path for the proposed Generalized ℓ_p-KTA formulation. Note that generating the entire path for the largest data set (Dorothea) is well beyond the scaling capabilities of all existing algorithms while our proposed algorithm generated the path in only half an hour on standard hardware; (b) plot validating our weak Generalized ℓ_p-KTA monotonicity conjecture that the number of selected features (kernels) decreases monotonically with p; (c) & (d) plots validating our strong Generalized ℓ_p-KTA monotonicity conjecture that feature (kernel) weights themselves vary monotonically beyond juncture points but vary non-monotonically in between (e). Figure best viewed in color under magnification.

Contributions

We propose a novel conjecture that, for certain MKL formulations subject to $p \geq 1$ regularization, the number of kernels (features) selected in the optimal solution monotonically decreases as p is decreased from an initial value to unity. We first prove that the conjecture is true for a generic family of Kernel Target Alignment (KTA) based ℓ_p-MKL formulations, henceforth termed as Generalized ℓ_p-KTA. This implies that regulating p, in such formulations, provides a principled way of generating the kernel selection path (see Fig 1(a), 1(b)). In fact, for this family, we further strengthen the conjecture and prove that the kernel weights themselves decay (grow) monotonically once they are below (above) a certain threshold at optimality (see Fig 1(c)-1(e)). This implies that there exist juncture points along the path and an algorithm can exploit these by eliminating certain kernels from the optimization for potentially large intervals of p. It should be noted that these conjectures are non-trivial and we show that they do not hold for the popular square loss based ℓ_p-MKL formulation.

Generalized ℓ_p-KTA formulation is strongly convex and leads to robust kernel selection [25, 59, 79]. The kernel monotonicity results allow us to develop a predictor-corrector based path following algorithm for Generalized ℓ_p-KTA that exploits the presence of juncture points for increased efficiency. We demonstrate the efficacy of the proposed formulation as well as the kernel selection path algorithm in the area of non-linear feature selection.

Non-linear Feature Selection

In non-linear feature selection [133, 89, 11, 65, 117], the predictor is a non-linear function of the input features. In many real world applications, one needs to determine the entire feature selection path so as to determine the most feasible operating point on the path in the context of the given application — motivated by considerations of elimination of noisy, expensive and redundant features, model compression for learning and predicting on a budget, model interpretability, *etc*. This is a challenge since state-of-the-art non-linear feature selection algorithms remain computationally expensive and training them thousands of times can be prohibitive (even with warm restarts). Multiple Kernel Learning (MKL) techniques have been shown to be amongst the most effective for non-linear feature selection [74, 32, 126, 127, 86, 64]. However, even though many specialized MKL optimization techniques have been developed [128, 104, 103, 81, 69], training them thousands of times with different parameter settings is often infeasible.

We perform extensive experiments on benchmark data sets to demonstrate that the proposed Generalized ℓ_p-KTA formulation can lead to better classification accuracies not only as compared to other ℓ_p-MKL formulations and uniform kernel baselines but also leading feature selection methods. We further demonstrate that our path following algorithm reduces training time significantly over other path following algorithms, state-of-the-art ℓ_p-MKL optimizers such as SMO-MKL, warm restarts and predictor-corrector baselines. In particular, we generate the entire feature selection path for data sets with a hundred thousand features in approximately half an hour on standard hardware. Entire path generation for such data set is well beyond the scaling capabilities of other methods.

Outline

The rest of the chapter is organized as follows. Section 7.1 discusses the existing works related to non-linear feature selection, MKL as well as path following algorithms. In Section 7.2, we present the proposed Generalized ℓ_p-KTA formulation. The novel conjecture on monotonic kernel selection path is proposed and proved in Section 7.3. The details of the predictor-corrector based kernel path following algorithm for solving Generalized ℓ_p-KTA are provided in Sections 7.4. Experimental results are discussed in Section 7.5. We conclude by summarizing the work and the key contributions.

7.1 Related Work

Considerable progress has been made in the area of non-linear feature selection. A popular approach is to map the input features to a kernel-induced feature space while simultaneously performing feature selection in the original input space. For example, [89] propose to perform LASSO regression in the induced space and thereby perform non-linear feature selection while [135, 54] pose this problem as that of tuning the hyper-parameters of the kernel. Another promising direction is to find un-correlated or independent features in the kernel induced feature space. While [138, 27, 137] aim at selecting orthogonal features in the feature space, [32, 117] employ Hilbert-Schmidt Independence Criterion based measures. Most of these approaches are either computationally expensive or resort to approximately minimizing their non-convex objectives.

Multiple Kernel Learning based techniques for non-linear feature selection have been explored in settings such as multi-label classification [74], bio-informatics [32, 86] and object categorization [127, 64]. In [126], MKL techniques for non-linear feature selection were shown to be better than boosting [15], lasso [6], sparse SVM [30], LP-SVM [52] and BAHSIC [117]. Various MKL formulations have been developed including $\ell_{p\geq 1}$-norm regularization over the kernel weights [83, 108, 39, 80, 42, 104, 103], mixed-norm regularizers [1], non-linear combinations of base kernels [40, 126], Bregman divergence based regularizers [128] and for regularized kernel discriminant analysis [140]. State-of-the-art $\ell_{p\geq 1}$-MKL optimization techniques such as SMO-MKL [128] and SPG-GMKL [69] have been shown to scale to a million kernels. Nevertheless, these techniques take more than a day to train on a standard desktop and so computing them thousands of times for determining the entire kernel selection path is infeasible.

Path following over the regularization parameter has been studied in the context of both non-sparse regularization [60] as well as sparse linear classifiers [154]. Some of the other settings where path following has been studied are: ℓ_1-MKL [13], general non-linear regularization paths [112], boosting [149] and ℓ_1 regularized feature selection [88]. Tracing the solution path for other hyper-parameters has also been explored. While [57, 129] perform path following over the tube width parameter in support vector regression, [130] follow the path for the kernel hyper-parameter in SVMs.

7.2 Generalized ℓ_p-norm Kernel Target Alignment

We introduce our Generalized ℓ_p-KTA formulation for sparse kernel learning using the following notation. Let $k^1, \ldots, k^n$ denote the given base kernel functions (one per feature) and let $K^1, \ldots, K^n$ denote the corresponding centered gram matrices obtained from the training data so that the features induced in the RKHS have zero mean [41]. Let $\mathbf{y}$ denote the vector with entries as the labels of the training data. We are interested in learning a kernel k that is a conic combination of the given base kernels: $k = \sum_{i=1}^{n} \eta_i k^i,\ \eta_i \geq 0\ \forall\ i = 1, \ldots, n.$ We focus on the following family of Bregman divergence based ℓ_p-KTA formulations for learning the kernel weights η.

Let F be a strictly convex and differentiable function and let ∇F denote its gradient. Then, the Bregman divergence generated by the function F is given by $B_F(x) = F(x) - F(x_0) - (x - x_0)^\top \nabla F(x_0)$, where x_0 is some fixed and given point in the domain of F. As an example, $F(x) = \langle x, x \rangle$ leads to $B_F(x) = \|x - x_0\|^2$, the squared Euclidean distance. The proposed Generalized ℓ_p-KTA formulations have the following form:

$$\min_{\eta \geq 0}\ \lambda_1 \bar{B}_F(\eta) + \lambda_2 \sum_{i=1}^{n} \eta_i^p - \sum_{i=1}^{n} \eta_i \mathbf{y}^\top K^i \mathbf{y}, \tag{7.1}$$

where the first term representing the Bregman divergence based regularizer is decomposable as $\bar{B}_F(\eta) = \sum_i B_F(\eta_i)$, the second term is the sparsity inducing ℓ_p regularizer and the third term captures the alignment of the learnt kernel to the ideal kernel $\mathbf{y}\mathbf{y}^\top$. Note that $p \geq 1,\ \lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are regularization parameters.

A large number of popular Bregman divergences such as the squared Euclidean distance, generalized KL-divergence, *etc.*, are decomposable [16] and hence admissible under the Generalize ℓ_p-KTA formulation. Such Bregman divergences are known to promote robust feature and kernel selection [25, 59, 79] while the ℓ_p-norm regularizer achieves variable sparsity [81]. Note that the classical KTA formulations studied in [83, 44] can be obtained as special cases of our formulation by taking F to be the squared Euclidean distance and having $\lambda_2 = 0$. On the other hand, substituting $\lambda_1 = 0$ results in the KTA formulation studied in Proposition (7) in [41]. The normalized KTA formulation in Proposition (9) in [41] employs a non-decomposable Bregman divergence and hence is not a special case of (7.1). However we empirically demonstrate in Section 7.5 that the proposed Generalized ℓ_p-KTA formulation outperforms this normalized

KTA formulation both in terms of generalization accuracy as well as computational cost. In particular, the normalized KTA formulation needs to operate on an $n \times n$ dense matrix which becomes infeasible for large n ($n = 10^5$ for the standard Dorothea dataset).

To perform non-linear feature selection, we associate a non-linear base kernel with each individual feature, such as RBF kernels defined per feature, and then learn a sparse combination of base kernels. The final classification results are obtained by training an SVM using the learnt kernel $k = \sum_{i=1}^{n} \eta_i k^i$. This corresponds to non-linear feature selection since features for which η_i is zero do not contribute to the kernel and can be dropped from the data set.

We conclude this section with the following observation. Though (7.1) optimizes the kernel weights independently for a given λ_1 and λ_2, the proposed algorithm employs cross-validation for optimizing λ_2. Hence, the final optimal kernel weights need not be independent.

7.3 Monotonic Kernel Selection Path

In this section, we prove the monotonicity conjecture for the proposed Generalized ℓ_p-KTA formulation. Let $\eta_i^*(p')$ denote the optimal weight of the kernel k^i obtained with (7.1) at $p = p'$ and let $\mathcal{K}(p')$ denote the set of kernels active at $p = p'$. A kernel k^i is said to be active at $p = p'$, i.e. $k^i \in \mathcal{K}(p')$. if and only if $\eta_i^*(p') > \epsilon$, where $\epsilon > 0$ is a user-defined tolerance. The proposed conjecture can now be formally stated as

$$k^j \notin \mathcal{K}(p') \Rightarrow k^j \notin \mathcal{K}(p) \; \forall \; 1 < p < p' \tag{7.2}$$

We begin our analysis by noting the following theorem:

Theorem 7.3.1. *The path of the optimal solutions of (7.1) with respect to p, i.e. $\eta^*(p)$, is unique, smooth (provided F is twice-differentiable) and, in general, non-linear.*

Proof is provided in appendix A.3.1.

The next theorem states the key result that proves the monotonicity conjecture for the proposed Generalized ℓ_p-KTA formulation.

Theorem 7.3.2. *Given $\eta_i^*(p')$, the following holds as p decreases from p' to unity: (1) $\eta_i^*(p)$ decreases monotonically whenever $\eta_i^*(p') < e^{-1}$; (2) $\eta_i^*(p)$ increases monotonically whenever $\eta_i^*(p') > e^{-\frac{1}{p'}}$*

Proof is provided in appendix A.3.2.

Theorem 7.3.2 implies that the monotonicity conjecture for the Generalized ℓ_p-KTA formulation in (7.1) holds whenever the user defined tolerance parameter ϵ (above which a kernel is said to be active) is set to be less than e^{-1}. Points wherever an optimal kernel weight becomes less than the threshold are referred to as juncture points. Note that Theorems 7.3.1,7.3.2 and the monotonicity conjecture are non-trivial as they do not hold for all loss functions. In particular, we analyze ℓ_p-MKL formulations with the popular square loss in Appendix A.3.3, and present settings where the conjecture does not hold. This implies that path following algorithms for such formulations cannot be speeded up by eliminating features whose weights touch zero from the optimization since they can increase to become non-zero at a later stage (see Figure A.1).

7.4 Efficient Path Following Algorithm

In this section, we present an efficient path following algorithm which closely approximates the true kernel selection path. The algorithm exploits the presence of juncture points to improve upon the standard Predictor-Corrector (PC) technique and scales effortlessly to problems involving a hundred thousand kernels. Finally, we give a bound on the deviation from the optimal objective function value caused due to the approximation.

Theorem 7.3.1 states that the solution path of (7.1) is smooth but non-linear in general. Path following is typically implemented using the standard Predictor-Corrector algorithm [2] in such cases [13, 112]. The PC algorithm is initialized with the optimal solution at $p = p_0$. At every iteration, the current value of p is decreased by a small step size, Δp and the following key iterative steps are performed:

Predictor: The predictor is a close approximation of $\eta_i^*(p - \Delta p)$ given $\eta_i^*(p)$. This can either be the warm start approximation, i.e. $\eta_i^*(p-\Delta p) = \eta_i^*(p)$, the first order approximation, $\eta_i^*(p-\Delta p) = \eta_i^*(p) - \Delta p \frac{\mathrm{d}\eta_i^*(p)}{\mathrm{d}p}$ or the second order approximation $\eta_i^*(p - \Delta p) = \eta_i^*(p) - \Delta p \frac{\mathrm{d}\eta_i^*(p)}{\mathrm{d}p} + \frac{1}{2}(\Delta p)^2 \frac{\mathrm{d}^2\eta_i^*(p)}{\mathrm{d}p^2}$ (the derivative expressions for our case are provided in Appendix A.3.4).

Corrector: The Newton method is used to correct the approximations of the predictor step leading to a quadratic convergence rate [2].

We modify the standard PC algorithm so as to closely approximate the solution path at

Algorithm 5 Generic Algorithm for Computing Solution Path of the Kernel Weights

```
**Input:** $\mathbf{y}^\top K^1\mathbf{y}, \ldots, \mathbf{y}^\top K^n\mathbf{y}$, step size $\Delta p(> 0)$, tolerance $\epsilon(> 0)$, start and end $p$ values: $p_0$ and $p_e$ respectively with $p_0 > p_e$.
**Output:** $\eta^*(p)$ for all p at $\Delta$ intervals between $[p_e, p_0]$
Initialization: $p = p_0$, Index set of active kernels $\mathcal{K} = \{1, \ldots, n\}$
Solve (7.1) at $p = p_0$ to obtain the optimal solution $\eta^*(p_0)$
if $(\eta_i^*(p_0)) < \epsilon$ for any $i \in \mathcal{K}$ then
    Set $\eta_i^*(p_0) = 0$
    Update $\mathcal{K} = \mathcal{K} \setminus \{i\}$
end if
repeat
    Update $p = p - \Delta p$
    Set $\eta_i^*(p) = 0$ for all $i \notin \mathcal{K}$
    for $i \in \mathcal{K}$ do
        Predictor: Initialize $\eta_i^*(p)$ using warm start or first or second order approximation
        Corrector: Run Newton's method to obtain $\eta_i^*(p)$
    end for
    if $(\eta_i^*(p)) < \epsilon$ for any $i \in \mathcal{K}$ then
        Set $\eta_i^*(p) = 0$
        Update $\mathcal{K} = \mathcal{K} \setminus \{i\}$
    end if
until $p \leq p_e$
```

a significantly lower computational cost. The key idea is to exploit the monotonicity in the active set size and directly set certain kernel weights to zero for all the subsequent p values. Algorithm 5 summarizes the proposed algorithm. The algorithm maintains an active-set that is initialized to those kernels whose optimal weight at p_0 is above ϵ. At every iteration, the first order PC step is used to determine the kernel weights for the next p value. Kernels whose weight falls below ϵ are eliminated from the active set for the remainder of the optimization due to the monotonicity property. The error incurred by the proposed method can be bounded in terms of ϵ. The following Lemma holds when the squared Euclidean distance is used as the Bregman divergence:

Lemma 7.4.1. *For any p, the deviation in the objective value of (7.1) obtained using the approximate path following algorithm from the true optimal objective is upper bounded by*

$n(\lambda_1\epsilon^2 + \lambda_2(p-1)\epsilon^p)$.

This result follows from the optimality conditions (A.17) and Theorem 7.3.2. We also state an analogous lemma for the generalized KL-divergence in Appendix A.3.5.

7.5 Experiments

We carry out experiments to determine both the classification accuracy of the proposed Generalized ℓ_p-KTA formulation as well as the computational cost of the proposed approximate path following algorithm.

Data sets & kernels: We present results on a variety of feature selection datasets taken from the NIPS 2003 Feature Selection Challenge [58] and the ASU Feature Selection Repository [150]. Table 7.1 lists the number of instances and features in each dataset. Unless otherwise stated, results are averaged via 5-fold cross-validation (standard deviations are reported in appendix A.3.6). We define an RBF kernel per feature as our base kernels and center and trace normalize them as recommended in [41].

Baseline techniques: We compare the proposed approach (Gen ℓ_p-KTA) to a number of baseline techniques including state-of-the-art Centered KTA formulation [41], highly optimized ℓ_p-MKL techniques such as SMO-MKL [128], leading feature selection methods such as BAHSIC [117] as well as path following approaches for ℓ_1-MKL (PF-ℓ_1-MKL) [13]. While evaluating classification accuracy, we also compare to the uniform kernel combination baseline ($\eta_i = 1/n \ \forall \ i$) referred to as Uniform as well as path following linear feature selection approaches (PF-ℓ_1-SVM) [154]. We also compare the proposed approach to 8 other state-of-the-art feature selection techniques whose details are given in [150]. The final classification results for all the feature selection algorithms are obtained using an SVM with the kernel com-

Data set	Num	Dim	Data set	Num	Dim
Arcene	100	10000	Madelon	2000	500
Relathe	1427	4322	Pcmac	1943	3289
Basehock	1993	4862	Dorothea	800	100000

Table 7.1: Data set statistics.

	Arcene	Madelon	Relathe	Pcmac	Basehock	Dorothea
Gen ℓ_p-KTA	$\mathbf{92.00}$ (3788)	$\mathbf{65.70}$ (12)	$\mathbf{92.57}$ (3644)	$\mathbf{93.62}$ (2947)	$\mathbf{98.59}$ (4262)	$\mathbf{94.75}$ (20220)
Centered-KTA	75.00 (134)	62.45 (290)	90.40 (542)	93.05 (498)	97.29 (584)	-
SMO-MKL	82.00 (9999)	62.05 (1)	-	-	-	-
BAHSIC	69.00 (100)	53.90 (50)	85.07 (500)	89.55 (500)	93.58 (500)	90.63 (500)
PF-ℓ_1-MKL	81.00 (87)	62.76 (89)	85.67 (287)	-	-	-
PF-ℓ_1-SVM	77.00 (273)	61.25 (7)	89.00 (510)	90.68 (190)	97.24 (264)	93.88 (499)
Uniform	81.00 (10000)	59.85 (500)	90.96 (4322)	92.49 (3289)	97.99 (4862)	91.38 (100000)

Table 7.2: The maximum classification accuracy achieved (with the corresponding number of selected features) along the kernel selection path. Generalized ℓ_p-KTA achieves significantly higher accuracies as compared to state-of-the-art KTA and $\ell_{p\geq 1}$-MKL formulations as well as leading feature selection techniques such as BAHSIC. The table reports mean results averaged over 5-fold cross validation (see the supplementary material for standard deviations). '-' denote results where the data set was too large for the feature selection algorithm to generate results.

puted as $k = \sum_{i=1}^{r} \eta_i k^i$, where $(\eta_i)_{i=1,\ldots,r}$ are the feature weights provided by the algorithm. While evaluating computational cost, we also compare the cost of our proposed approximate first order predictor-corrector algorithm to the exact path following algorithm.

Parameter settings: For the Generalized ℓ_p-KTA formulation we set λ_1, λ_2 and the SVM misclassification penalty C by 5-fold cross-validation while varying p from 2 to 1 in decrements of 0.01. We use the squared Euclidean distance as the Bregman divergence with $\mathbf{x}_0 = 0$. BAHSIC, PF-ℓ_1-MKL and PF-ℓ_1-SVM are parameter free when computing the kernel selection path. The parameters of the other techniques were also set via extensive cross-validation except for the computationally intensive SMO-MKL where this was feasible only for the smaller data sets. On the larger data sets. we follow the SMO-MKL authors' experimental protocol and fix $\lambda = 1$ and validate over C with $p \in \{1.01, 1.33, 1.66, 2\}$. As in the case of previous ℓ_p-MKL algorithms [128, 104, 103, 69], we employ the common strategy of thresholding to obtain sparse solutions for the proposed Generalized ℓ_p-KTA.

Results: Table 2 lists the maximum classification accuracy achieved along the entire kernel selection path and the corresponding number of selected features (*i.e.*, corresponding to the best oracle results on the test set). Our proposed Generalized ℓ_p-KTA formulation gets significantly higher classification accuracies than all competing methods. For instance, on the Arcene data set, Generalized ℓ_p-KTA achieves a classification accuracy which is 11% higher than the closest competing method.

	Arcene	Madelon	Relathe	Pcmac	Basehock	Dorothea
Gen l_p-KTA (RBF)	76.80 (124)	64.50 (14)	89.40 (183)	89.76 (196)	95.46 (184)	93.75 (17)
Gen l_p-KTA (Linear)	**73.40** (16)	62.04 (12)	**88.39** (190)	88.88 (180)	94.76 (189)	**93.60** (12)
Inf. Gain	72.00 (110)	61.63 (5)	84.39 (190)	**88.99** (135)	95.26 (200)	93.33 (35)
Chi-Square	71.20 (120)	61.69 (10)	83.48 (180)	88.24 (155)	**95.28** (160)	93.33 (20)
Fisher Score	66.20 (65)	61.47 (10)	83.35 (180)	88.02 (100)	94.61 (200)	93.30 (20)
mRMR	68.20 (60)	61.87 (5)	75.01 (60)	83.34 (145)	88.88 (70)	93.18 (155)
ReliefF	68.40 (170)	**62.06** (15)	77.08 (200)	80.76 (200)	86.05 (200)	93.33 (105)
Spectrum	64.00 (195)	60.19 (25)	69.99 (175)	66.74 (185)	69.79 (200)	90.28 (140)
Gini Index	64.60 (185)	59.43 (25)	69.50 (180)	66.60 (185)	69.49 (200)	90.28 (140)
K.-Wallis	60.20 (95)	55.04 (65)	70.97 (200)	65.20 (200)	70.37 (200)	90.08 (65)

Table 7.3: The maximum classification accuracy achieved on the ASU data sets (with the corresponding number of selected features) along the kernel selection path. In keeping with the ASU experimental protocol, all algorithms are restricted to selecting at most 200 features and are allowed to train on only half the data. Generalized ℓ_p-KTA (RBF) outperforms all the linear techniques and this demonstrates the advantages of non-linear feature selection. Amongst the linear methods, our proposed method with linear features is the best in general.

Table 7.3 presents results on the ASU data sets following the ASU experimental protocol where only 50% of the data is used for training and the number of selected features is restricted to be less than 200. This tests the capabilities of feature selection algorithms under the demanding conditions of both limited training data and limited prediction budget. As can be seen, Generalized l_p-KTA (RBF) clearly outperforms all the linear methods thereby demonstrating the power of non-linear feature selection. Amongst all the linear feature selection methods, our proposed Gen l_p-KTA (Linear) is the best in general. It is the best method on 3 data sets – on Relathe it is better than the second best method by 4%, on Arcene by 1.4% and on Dorothea by 0.27%. On 2 data sets it is the second best method – lagging behind the best method on Madelon by 0.02% and on Pcmac by 0.11%. These results demonstrate that the Generalized ℓ_p-KTA formulation can lead to better results not only as compared to other KTA and ℓ_p-MKL formulations but also as compared to leading feature selection techniques.

Table 7.4 assesses the cost of computing the kernel selection path. Note that algorithms such as BAHSIC compute the path a feature at a time while other algorithms, such PF-ℓ_1-MKL, have a much sparser sampling of the path (in the limit Centered KTA produces only a single point on the path). Therefore, for a fair comparison, we report the total time taken by each algorithm divided by the number of points generated by it along the path. This measures the average time taken by each algorithm to generate a point along the path. Table 7.4 shows that

	Arcene	Madelon	Relathe	Pcmac	Basehock	Dorothea
Gen ℓ_p-KTA	0.3	2.4	2.4	3.7	4.8	12.5
Centered-KTA	10.9	2405.5	2093.8	1099.6	3302.6	-
SMO-MKL	5.7	168.4	-	-	-	-
BAHSIC	116.2	2265.8	3764.2	5906	7847.1	10976.3
PF-ℓ_1-MKL	29.5	83.1	240.0	-	-	-

Table 7.4: Average training time (in seconds) required to compute a point along the kernel selection path. Note that each algorithm generates a different number of points along the path. Therefore, for a fair comparison, we report the total time taken to compute the path for each algorithm divided by the number of points generated by the algorithm. Our path following algorithm for the Generalized ℓ_p-KTA formulation is orders of magnitude faster than competing techniques and is the only algorithm which can generate the entire path for the Dorothea data set. '-' denote results where the data set was too large for the feature selection algorithm to generate results.

the proposed approximate first order predictor-corrector algorithm can be significantly faster than competing techniques. Furthermore, on the largest data set, Dorothea with 10^5 features, Centered-KTA, SMO-MKL and PF-ℓ_1-MKL were not able to generate even a single point on the path whereas our algorithm was able to generate the entire path in 21 minutes using pre-computed kernels and 36 minutes using kernels computed on the fly. BAHSIC generates the path by adding a single feature at each iteration and was able to generate the initial path segment up to 500 features but at a high computational cost of more than 3 hours. All experiments were carried out on a standard 2.40 GHz Intel Xeon desktop with 32 GB of RAM. Finally, we note that our approximate path following algorithm was also found to be faster than the exact path following algorithm implemented with first order predictor or with warm restarts without exploiting juncture points and eliminating features from the optimization. The speedups against both the baselines varied with more than 3 times on Madelon and almost 4 times on Dorothea while the maximum relative deviation from the exact path was only 7.3×10^{-6} and 3.6×10^{-4} respectively.

7.6 Conclusions

We developed an efficient kernel selection path algorithm for a generic family of kernel target alignment based ℓ_p-MKL formulation. Our starting point was a novel conjecture that the number of selected kernels, and the kernels weights themselves, vary monotonically with p in ℓ_p-MKL formulations. We proposed a Generalized ℓ_p-KTA formulation and proved that the conjecture holds for this formulation but not for other popular formulations such as the square loss based ℓ_p-MKL. The monotonicity property of the Generalized ℓ_p-KTA formulation allows us to eliminate kernels whose weight goes below a certain threshold from the optimization for large intervals of p leading to efficient path following. It was theoretically and empirically demonstrated that this resulted in only a minor deviation from the exact path but achieved significant speedups. Experiments also revealed that our proposed formulation could yield significantly higher classification accuracies as compared to state-of-the-art MKL and feature selection methods and that our approximate path following algorithm could be orders of magnitude faster than other leading path following techniques. Future work includes investigating the solution path of other popular ℓ_p-MKL setting [81, 128] and developing efficient path following algorithms for them.

Chapter 8

Conclusions

In this final chapter, we summarize the main contributions of this thesis and discuss possible directions of future work.

8.1 Summary

The paradigm of training multiple related tasks jointly raises interesting challenges and presents new opportunities for learning the underlying models in unconventional ways. In this thesis, we focused on the problem of learning which feature information is to be shared among which tasks. A key idea explored by us is to pose the problem of learning the features shared among multiple tasks as that of learning a shared kernel. We proposed diverse models for multi-task kernel learning. In addition to the models developed, derivations of specialized dual formulations as well as the algorithms proposed for solving them efficiently are also an important contribution of this thesis.

Learning a common feature representation across all the tasks is a popular multi-task setting. In Chapter 3, we modeled the shared kernel as a linear combination of a given set of base kernels, the Multiple Kernel Learning (MKL) framework. A generic ℓ_q/ℓ_p mixed-norm formulation is proposed to learn the shared kernel as well as prediction functions. The q-norm, $q \in [1, 2)$, over the base kernels promotes sparsity in the kernel combination while the p-norm over the tasks induces varying degree of relatedness among the tasks. Efficient mirror descent and sequential minimal optimization (SMO) based algorithms were proposed to solve the proposed formulation. The formulation is far more scalable than existing multi-task feature learning methodologies [5, 4, 8, 33, 9]. Experiments on object categorization and other real world datasets showed that it achieves a good generalization when provided with a suitable set of base kernels.

Instead of sharing the feature spaces induced by given kernels, the tasks may share the latent feature spaces residing within those kernels. For such settings, we model the feature space induced by the shared kernel as a low-dimensional orthonormal transformation of the feature spaces induced by the base kernels. In Chapter 4, we pose this as a ℓ_1/ℓ_q $(q \geq 1)$ mixed Schatten-norm regularized optimization problem. Mixed Schatten-norm regularized problems are non-standard in literature and call for novel optimization methodologies. We showed that the negative entropy function can be employed as an efficient Bregmann generating function in the context of mirror-descent for solving such mixed Schatten-norm regularized problems. Simulations on benchmark real-world datasets showed that the proposed formulation achieved statistically significant improvement against state-of-the-art multi-task feature learning algorithm. The results also confirmed that in this setup, learning suitable kernels is essential for obtaining a good generalization — baselines with uniform kernel combination yielded worse performance than the baselines employing simple cross-validation over base kernels.

In Chapter 5, we modeled the shared kernel as a polynomial combination of base kernels. Such combinations may be encoded in a directed acyclic graph (DAG). The size of such DAGs can become exponential in the number of base kernels. We proposed to explore this DAG by employing a carefully designed ℓ_1/ℓ_p block-norm graph based regularizer that induces structure sparsity within the DAG and enables us to search it with an active set algorithm. Even when the DAG is exponential in size, the complexity of the proposed algorithm is polynomial in the size of active set. We also derive a sufficient condition of optimality for our algorithm. We illustrate the efficacy of the proposed formulation in the application of Rule Ensemble Learning (REL), which aims at learning an optimal ensemble of conjunctive rules. We model the conjunctive rules as a polynomial combination of a special set of base kernels and pose the problem of learning rule ensemble as a kernel learning problem. Empirical studies in REL setting showed that the proposed formulation learnt rules that are easier to interpret and have better generalization ability than the existing REL approaches.

In Chapter 6, we considered the setting where some relevant kernels (feature spaces) are shared across few tasks. We aim at searching the exponentially large space of all possible groups of tasks that may share a kernel. We propose a graph based multi-task regularizer that encodes the following important structure among the groups of tasks that leads to an efficient active set based algorithm for solving it: if a group of tasks do not share a kernel, then none of its superset share a kernel. Empirical results on benchmark datasets underline the importance of learning which kernels are shared by among which tasks. In particular, we see a case where learning the tasks independently performs much better than the multi-task learning algorithms that assume all the tasks to be related and/or have a common kernel (feature space) for all the tasks. However, our formulation achieves statistically significant improvement in this dataset. Overall, the proposed formulation achieves good generalization and outperforms state-of-the-art multi-task learning algorithms in many cases.

We explored the problem of computing the kernel selection path for p-norm Multiple Kernel Learning (MKL) framework in Chapter 7. Kernel selection path is the variation in the classification accuracy as the fraction of selected kernels is varied from null to unity. In ℓ_p-MKL setting, the fraction of selected kernels as well as the generalization achieved by the formulation varies significantly with the parameter p. We propose a novel conjecture which states that, for certain ℓ_p-MKL formulations, the number of features selected in the optimal solution monotonically decreases as p is decreased from an initial value to unity. We prove the conjecture for a generic family of kernel target alignment based ℓ_p-MKL formulations. This allows us to develop an efficient path computing algorithm for MKL. We empirically demonstrate that our formulation can lead to classification accuracies which are as much as 10% higher on benchmark data sets not only as compared to other ℓ_p-MKL formulations and uniform kernel baselines but also leading feature selection methods. We further demonstrate that our path following algorithm reduces training time significantly over other path following algorithms, state-of-the-art MKL optimizers such as SMO-MKL, warm restarts and predictor-corrector baselines. In particular, we generate the entire feature selection path for data sets with a hundred thousand features in approximately half an hour on standard hardware. Entire path generation for such data set is well beyond the scaling capabilities of other methods.

8.2 Future Work

Discovering the correlations existing among learning problems and leveraging them to obtain better performance is a central theme in machine learning. Our work primarily focused on kernel learning based feature induction in various multi-task settings. However, the relatedness among the tasks may not be limited to kernel (feature) sharing. For instance, in the area of inductive logic programming, one is typically provided with *background knowledge* for every problem (task). Hence, it may be the case that the tasks share the same *background knowledge* in such settings. A possible extension of our framework is to design algorithms that may exploit such prior knowledge during the training phase.

Tasks may also have temporal correlations. Such examples are common in the applications of computational biology and finance. [152, 131] studied the problem of predicting disease progression for several patients at successive time slots. They proposed models that learned the tasks corresponding to some given time slots. An interesting direction of research is to develop models that learn the prediction functions for future time slots as well. The framework of operator-valued kernels and functional data analysis [76] is a promising direction in this regard.

Limited work has been done in the area of structured prediction when multiple related tasks are involved. [53] have proposed a generic extension of the max-margin based structured output prediction formulation [123] to multiple tasks while [111] considered the case where the structured output (for each task) is a vector in finite dimensional Euclidean space. Developing novel formulations and efficient algorithms in this setting will be a useful contribution.

Our work in Rule Ensemble Learning is limited to the domain of propositional logic/features. It will be interesting if the proposed formulation is generalized to the First Order Logic (FOL) framework. This may involve exploring other feature induction paradigms such as those employed in FOL setups.

In Chapter 7, we showed that monotonic kernel selection path is not guaranteed for generic Multiple Kernel Learning (MKL) formulations. Investigating the nature of this path for other popular MKL settings and developing efficient algorithms for computing it in those setups is another interesting line of work.

Appendix A

A.1 Learning Multiple Tasks with Hierarchical Kernels

A.1.1 Lemma 26 of [95]

Let $a_i \geq 0, i = 1, \ldots, d$ and $1 \leq r < \infty$. Then, for $\Delta_{d,r} = \left\{\eta \in \mathbb{R}^d \mid \eta \geq 0, \sum_{i=1}^d \eta_i^r \leq 1\right\}$

$$\min_{\eta \in \Delta_{d,r}} \sum_{i=1}^{d} \frac{a_i}{\eta_i} = \left(\sum_{i=1}^{d} a_i^{\frac{r}{r+1}}\right)^{1+\frac{1}{r}}$$

and the minimum is attained at

$$\eta_i = \frac{a_i^{\frac{1}{r+1}}}{\left(\sum_{i=1}^d a_i^{\frac{r}{r+1}}\right)^{\frac{1}{r}}} \quad \forall\, i = 1, \ldots, d.$$

A.1.2 Proof of Lemma 5.2.1

Proof. Applying the above lemma (A.1.1) on the outermost ℓ_1-norm of $\Omega_T(f_1, \ldots, f_T)^2$ (5.3), we have

$$\Omega_T(f_1, \ldots, f_T)^2 == \min_{\gamma \in \Delta_{|V|,1}} \sum_{v \in V} \frac{d_v^2}{\gamma_v} \left(\sum_{w \in D(v)} (Q_w(f_1, \ldots, f_T))^\rho\right)^{\frac{2}{\rho}}$$

Reapply the above lemma on the individual terms of the above summation, we get

$$\left(\sum_{w \in D(v)} (Q_w(f_1, \ldots, f_T)^2)^{\frac{\rho}{2}}\right)^{\frac{2}{\rho}} = \min_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}}} \sum_{w \in D(v)} \frac{Q_w(f_1, \ldots, f_T)^2}{\lambda_{vw}}$$

Using the above two results and regrouping the terms will complete the proof. □

A.1.3 Re-parameterization of the Multi-task Regularizer (5.5)

The gHKL$_{\text{MT}}$ dual formulation (5.7) follows from the representer theorem [113] after employing the following reparameterization in (5.5).

Define $f^{0w} = \frac{1}{T+\mu}\sum_{t=1}^{T} f_{tw}$ and $f^{tw} = f_{tw} - f^{0w}$. Then, $Q_w(f_1,\ldots,f_T)$ may be rewritten as:

$$Q_w(f_1,\ldots,f_T) = \left(\mu\|f^{0w}\|^2 + \sum_{t=1}^{T}\|f^{tw}\|^2\right)^{\frac{1}{2}}$$

Further, construct the following feature map [48]

$$\Phi_w(\mathbf{x},t) = \Big(\ \frac{\phi_w(\mathbf{x})}{\sqrt{\mu}},\ \underbrace{\mathbf{0},\ldots,\mathbf{0}}_{\text{for tasks before t}}\ ,\phi_w(\mathbf{x}),\ \underbrace{\mathbf{0},\ldots,\mathbf{0}}_{\text{for tasks after t}}\ \Big) \tag{A.1}$$

and define $f_w = (\sqrt{\mu}f^{0w}, f^{1w},\ldots,f^{Tw})$.

With the above definitions, it can be observed that $Q_w(f_1,\ldots,f_T)^2 = \|f_w\|^2$ and $F_t(\mathbf{x}) = \sum_{w\in\mathcal{V}}\langle f_w, \Phi_w(\mathbf{x},t)\rangle - b_t\ \forall\ t$. It follows from Lemma 5.2.1 that the gHKL$_{\text{MT}}$ primal problem based on (5.5) is equivalent to:

$$\min_{\gamma\in\Delta_{|\mathcal{V}|,1}}\ \min_{\lambda_v\in\Delta_{|D(v)|,\hat{\rho}}\forall v\in V}\ \min_{f,b}\ \frac{1}{2}\sum_{w\in\mathcal{V}}\delta_w(\gamma,\lambda)^{-1}\|f_w\|^2 + C\sum_{t=1}^{T}\sum_{i=1}^{m}\ell(y_{ti},F_t(\mathbf{x}_{ti})) \tag{A.2}$$

where $f = (f_w)_{w\in\mathcal{V}}$ and $b = [b_1,\ldots,b_T]$.

A.1.4 Min-max interchange

Claim: The problem (5.7) remains the same whether solved with the original set of variables (γ,λ) or when solved with only those $\gamma_v \neq 0$ and $\lambda_{vw}\neq 0$ at optimality.

Proof: It follows from the following arguments: a) variables γ and λ owe their presence in (5.7) only via $\delta(\gamma,\lambda)$ functions, b) $(\gamma_v = 0) \Rightarrow (\delta_w(\gamma,\lambda) = 0\ \forall w\in D(v))$ and $(\exists v\in A(w)|\lambda_{vw} = 0) \Rightarrow (\delta_w(\gamma,\lambda) = 0)$, and c) min-max interchange in (5.7) yields an equivalent formulation. Its proof is provided in the following lemma.

Lemma A.1.1. *The following min-max interchange is equivalent*

$$\min_{\gamma\in\Delta_{|\mathcal{V}|,1}}\ \min_{\lambda_v\in\Delta_{|D(v)|,\hat{\rho}}\forall v\in\mathcal{V}}\ \max_{\alpha_t\in S(\mathbf{y}_t,C)\forall t} G(\gamma,\lambda,\alpha) = \max_{\alpha_t\in S(\mathbf{y}_t,C)\forall t}\ \min_{\gamma\in\Delta_{|V|,1}}\ \min_{\lambda_v\in\Delta_{|D(v)|,\hat{\rho}}\forall v\in V} G(\gamma,\lambda,\alpha)$$

where $G(\gamma,\lambda,\alpha)$ is as defined in (5.7).

Proof. We proceed by applying a change of variables. Note that $\gamma_v = 0$ implies that the variables λ_{vw} ($\forall w \in D(v)$) do not influence the objective of optimization problem (5.7). This follows from the definition of the $\delta(\gamma, \lambda)$ function. Hence, we define $\beta_{vw} = \gamma_v \lambda_{vw},\ \forall w \in D(v)$ as it is a one-to-one transformation for $\gamma_v \neq 0$ [see also 120]. The gHKL dual (5.7) (the L.H.S. of the proposed lemma) can be equivalently rewritten as

$$\min_{\substack{\beta_{vw} \geq 0\ \forall w \in D(v), v \in \mathcal{V} \\ \sum_v \|\beta_{vD(v)}\|_{\hat{\rho}} \leq 1}} \quad \max_{\alpha_t \in S(\mathbf{y}_t, C) \forall t} G(\beta, \alpha) \quad \text{where } \beta_{vD(v)} = (\beta_{vw})_{w \in D(v)},$$

$$G(\beta, \alpha) = \mathbf{1}^\top \alpha - \frac{1}{2} \alpha^\top \mathbf{Y} \left(\sum_{w \in \mathcal{V}} \delta_w(\beta) H_w \right) \mathbf{Y} \alpha \text{ and } \delta_w(\beta)^{-1} = \sum_{v \in A(w)} \frac{d_v^2}{\beta_{vw}}$$

Note that $\delta_w(\beta)$ is a concave function of β (in the given feasibility set) and hence $G(\beta, \alpha)$ is convex-concave function with convex and compact feasibility sets. Therefore, applying the Sion-Kakutani minmax theorem [116], we have $\min_\beta \max_\alpha G(\beta, \alpha) = \max_\alpha \min_\beta G(\beta, \alpha)$ (with constraints over β and α as stated above). Finally, we revert to the original variables (γ, λ): $\gamma_v = (\sum_{w \in D(v)} (\beta_{vw})^{\hat{\rho}})^{\frac{1}{\hat{\rho}}}\ \forall\, v \in \mathcal{V}$ and $\lambda_{vw} = \frac{\beta_{vw}}{\gamma_v}\ \forall\, w \in D(v),\ \forall\, v \in \mathcal{V}$ s.t. $\gamma_v \neq 0$. This gives us the equivalent

$$\max_{\alpha_t \in S(\mathbf{y}_t, C) \forall t} \quad \min_{\gamma \in \Delta_{|V|,1}} \quad \min_{\lambda_v \in \Delta_{|D(v)|, \hat{\rho}} \forall v \in V} G(\gamma, \lambda, \alpha)$$

□

A.1.5 Proof of Theorem 5.2.3

Before stating the proof of Theorem 5.2.3, we first prove the following results that will be employed therein (also see Proposition 11 of [10]).

Lemma A.1.2. *Let* $a_i > 0\, \forall\, i = 1, \ldots, d$, $1 < r < \infty$ *and* $\Delta_{d,1} = \left\{ \mathbf{z} \in \mathbb{R}^d \mid \mathbf{z} \geq 0, \sum_{i=1}^d \mathbf{z}_i \leq 1 \right\}$. *Then*

$$\min_{\mathbf{z} \in \Delta_{d,1}} \sum_{i=1}^d a_i \mathbf{z}_i^r = \left(\sum_{i=1}^d a_i^{\frac{1}{1-r}} \right)^{1-r}$$

and the minimum is attained at

$$\mathbf{z}_i = a_i^{\frac{1}{1-r}} \left(\sum_{j=1}^d a_i^{\frac{1}{1-r}} \right)^{-1} \quad \forall\, i = 1, \ldots, d.$$

Proof. Take vectors $\mathbf{u}_1$ and $\mathbf{u}_2$ as those with entries $a_i^{\frac{1}{r}}\mathbf{z}_i$ and $a_i^{-\frac{1}{r}}$ $\forall\, i = 1, \ldots, d$ respectively. The result follows from the Holder's inequality: $\mathbf{u}_1^\top \mathbf{u}_2 \leq \|\mathbf{u}_1\|_r \|\mathbf{u}_2\|_{\frac{r}{r-1}}$. Note that if any $a_i = 0$, then the optimal value of the above optimization problem is zero. □

Proposition A.1.3. *The following convex optimization problems are dual to each other, and there is no duality gap:*

$$\max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) M_w, \tag{A.3}$$

$$\min_{\kappa \in L} \max_{u \in \mathcal{V}} \sum_{w \in D(u)} \frac{\kappa_{uw}^2 \lambda_{uw} M_w}{d_u^2} \tag{A.4}$$

where $L = \{\kappa \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|} \mid \kappa \geq 0,\ \sum_{v \in A(w)} \kappa_{vw} = 1,\ \kappa_{vw} = 0\ \forall\, v \in A(w)^c,\ \forall\, w \in \mathcal{V}\}$, $\Delta_{n,r} = \{\mathbf{z} \in \mathbb{R}^n \mid \mathbf{z} \geq 0, \sum_{i=1}^n \mathbf{z}_i^r \leq 1\}$ *and* $M_w \geq 0\ \forall\, w \in \mathcal{V}$.

Proof. The optimization problem (A.4) may be equivalently rewritten as:

$$\min_{\kappa \in L} \min_{A} A \quad \text{subject to } A \geq \sum_{w \in D(u)} \frac{\kappa_{uw}^2 \lambda_{uw} M_w}{d_u^2}\ \forall\, u \in \mathcal{V}$$

$$= \min_{\kappa \in L} \max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \sum_{u \in \mathcal{V}} \sum_{w \in D(u)} \frac{\gamma_u \kappa_{uw}^2 \lambda_{uw} M_w}{d_u^2} \qquad \text{(Lagrangian dual with respect to } A\text{)}$$

$$= \max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \min_{\kappa \in L} \sum_{w \in \mathcal{V}} \left(\sum_{u \in A(w)} \kappa_{uw}^2 \frac{\gamma_u \lambda_{uw}}{d_u^2} \right) M_w \qquad \text{(min-max interchange and rearranging terms)}$$

$$= \max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \sum_{w \in \mathcal{V}} \left(\sum_{u \in A(w)} \left(\frac{\gamma_u \lambda_{uw}}{d_u^2} \right)^{-1} \right)^{-1} M_w \qquad \text{(Lemma A.1.2 with respect to variables } \kappa\text{)}$$

$$= \max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) M_w$$ □

We now begin with the proof of Theorem 5.2.3.

Proof. From Lemma A.1.1, the gHKL dual (5.7) can be equivalently written as

$$\max_{\alpha \in S(\mathbf{y}, C)} \mathbf{1}^\top \alpha - \frac{1}{2} \underbrace{\max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \max_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} \left(\sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) \alpha^\top \mathbf{Y} H_w \mathbf{Y} \alpha \right)}_{\mathcal{O}} \tag{A.5}$$

Here $\hat{\rho} = \frac{\rho}{2-\rho}$. In the following, we equivalently rewrite the second part of the above formulation

$$
\begin{aligned}
\mathcal{O} &= \max_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} \ \max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) \underbrace{\alpha^\top \mathbf{Y} H_w \mathbf{Y} \alpha}_{M_w} \\
&= \max_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} \ \min_{\kappa \in L} \ \max_{u \in \mathcal{V}} \sum_{w \in D(u)} \frac{\kappa_{uw}^2 \lambda_{uw} M_w}{d_u^2} \qquad \text{(Proposition A.1.3)} \\
&= \max_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} \ \min_{\kappa \in L} \ \min_{A} \ A \qquad \text{(Eliminating } u\text{)} \\
&\qquad \text{s.t. } A \geq \sum_{w \in D(v)} \frac{\kappa_{vw}^2 \lambda_{vw} M_w}{d_v^2} \ \forall \, v \in \mathcal{V} \\
&= \min_{\kappa \in L} \ \min_{A} \ \max_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} \ A \qquad \text{(Sion-Kakutani theorem)} \\
&\qquad \text{s.t. } A \geq \sum_{w \in D(v)} \frac{\kappa_{vw}^2 \lambda_{vw} M_w}{d_v^2} \ \forall \, v \in \mathcal{V} \\
&= \min_{\kappa \in L} \ \min_{A} \ A \qquad \text{(Holder's inequality, } \bar{\rho} = \tfrac{\hat{\rho}}{\hat{\rho}-1}\text{)} \\
&\qquad \text{s.t. } A \geq d_v^{-2} \left(\sum_{w \in D(v)} \left(\kappa_{vw}^2 M_w \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}} \ \forall \, v \in \mathcal{V} \\
&= \min_{\kappa \in L} \ \max_{u \in \mathcal{V}} \ d_u^{-2} \left(\sum_{w \in D(u)} \left(\kappa_{uw}^2 M_w \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}} \qquad \text{(Eliminating } A\text{)} \qquad \text{(A.6)}
\end{aligned}
$$

Now consider the problem $\mathcal{O}^{\bar{\rho}} = \min_{\kappa \in L} \max_{u \in \mathcal{V}} d_u^{-2\bar{\rho}} \sum_{w \in D(u)} \left(\kappa_{uw}^2 M_w \right)^{\bar{\rho}}$. Its lagrangian is

$$
\mathcal{L}(\kappa, A, \eta) = A + \sum_{v \in \mathcal{V}} \eta_v \left(d_v^{-2\bar{\rho}} \sum_{w \in D(v)} \left(\kappa_{vw}^2 M_w \right)^{\bar{\rho}} - A \right)
$$

Minimization of $\mathcal{L}$ with respect to A leads to the constraint $\eta \in \Delta_{|\mathcal{V}|,1}$. Hence

$$
\mathcal{O}^{\bar{\rho}} = \max_{\eta \in \Delta_{|\mathcal{V}|,1}} \ \min_{\kappa \in L} \sum_{v \in \mathcal{V}} \sum_{w \in D(v)} \eta_v \left(d_v^{-2} \kappa_{vw}^2 M_w \right)^{\bar{\rho}}
$$

Using the special structure of L, the above can be rewritten as:

$$
\mathcal{O}^{\bar{\rho}} = \max_{\eta \in \Delta_{|\mathcal{V}|,1}} \sum_{w \in \mathcal{V}} (M_w)^{\bar{\rho}} \left(\min_{\kappa_w \in \Delta_{|A(w)|,1}} \sum_{v \in A(w)} \left(\eta_v d_v^{-2\bar{\rho}} \right) \kappa_{vw}^{2\bar{\rho}} \right)
$$

Applying Lemma A.1.2 with respect to variables κ, we have that

$$
\min_{\kappa_w \in \Delta_{|A(w)|,1}} \sum_{v \in A(w)} \left(\eta_v d_v^{-2\bar{\rho}} \right) \kappa_{vw}^{2\bar{\rho}} = \zeta_w(\eta) = \left(\sum_{v \in A(w)} d_v^{\rho} \eta_v^{1-\rho} \right)^{\frac{1}{1-\rho}} \qquad \text{(A.7)}
$$

From the above two results, we obtain the following equivalent dual of (A.6)

$$\mathcal{O} = \max_{\eta \in \Delta_{|\mathcal{V}|,1}} \left(\sum_{w \in \mathcal{V}} \zeta_w(\eta) \left(\alpha^\top \mathbf{Y} H_w \mathbf{Y} \alpha \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}} \tag{A.8}$$

Substituting $\mathcal{O}$ in (A.5) by the above (A.8) and again interchanging the min-max completes the proof. □

A.1.6 Proof of Theorem 5.3.1

Proof. We begin by noting that $\zeta_v(\eta)$ $(v \in \mathcal{V})$ is a concave function of η for all v (this is because when $\rho \in (1,2]$, ζ_v is a weighted q-norm in η, where $q \in [-1,0)$ and hence is concave in the first quadrant). By simple observations regarding operations preserving convexity we have that the objective in (5.11) is a convex function of η for a fixed value of α. Hence $g(\eta)$, which is a point-wise maximum over convex functions, is itself convex. The expression for $\nabla g(\eta)$ is computed by employing the Danskin's theorem (prop. B.25 in [23]) and is as follows:

$$(\nabla g(\eta))_i = -\frac{(1-\varepsilon)}{2\bar{\rho}} \times \overbrace{\left(\sum_{u \in D(i)} d_i^{\rho} \left((1-\varepsilon)\eta_i + \frac{\varepsilon}{|\mathcal{V}|} \right)^{-\rho} \zeta_u^s(\eta)^{\rho} \left(\bar{\alpha}^\top \mathbf{Y} H_u \mathbf{Y} \bar{\alpha} \right)^{\bar{\rho}} \right)}^{P_1} \tag{A.9}$$
$$\times \underbrace{\left(\sum_{w \in \mathcal{V}} \zeta_w^s(\eta) \left(\bar{\alpha}^\top \mathbf{Y} H_w \mathbf{Y} \bar{\alpha} \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}-1}}_{P_2}$$

where $\bar{\rho} = \frac{\rho}{2(\rho-1)}$, $\zeta_w^s(\eta) = \zeta_w((1-\varepsilon)\eta + \frac{\varepsilon}{|\mathcal{V}|})$, *i*.e., the smoothed $\zeta_w(\eta)$ and $\bar{\alpha}$ is an optimal solution of problem (5.11) with that η where the gradient is to be computed.

Next, we show that g is Lipschitz continuous by showing that its gradient is bounded. Firstly, $\rho \in (1,2]$ and hence $\bar{\rho} \in [1,\infty)$. Next, let the minimum and maximum eigenvalues over all H_w $(w \in \mathcal{V})$ be θ and σ respectively. Then we have $\theta\|\bar{\alpha}\|^2 \leq \bar{\alpha}^\top \mathbf{Y} H_w \mathbf{Y} \bar{\alpha} \leq \sigma\|\bar{\alpha}\|^2$. Using this, we obtain: $\sum_{w \in \mathcal{V}} \zeta_w^s(\eta) \left(\bar{\alpha}^\top \mathbf{Y} H_w \mathbf{Y} \bar{\alpha} \right)^{\bar{\rho}} \geq \theta^{\bar{\rho}} \|\bar{\alpha}\|^{2\bar{\rho}} \sum_{w \in \mathcal{V}} \zeta_w^s(\eta)$. Note that $\sum_{w \in \mathcal{V}} \zeta_w^s(\eta) \geq \zeta_r^s(\eta)$ where $r \in \mathit{sources}(\mathcal{V})$ and $\zeta_r^s(\eta) \geq d_{max}^{\rho/(1-\rho)} \frac{\varepsilon}{|\mathcal{V}|}$ where d_{max} is the maximum of d_v $(v \in \mathcal{V})$. Thus we obtain: $P_2 \leq (\theta^{\bar{\rho}} \|\bar{\alpha}\|^{2\bar{\rho}} \varepsilon / |\mathcal{V}|)^{\frac{1}{\bar{\rho}}-1} d_{max}^{\frac{2-\rho}{\rho-1}}$.

Now, it is easy to see that $\forall\ u \in D(i)$, $d_i^{\rho}((1-\varepsilon)\eta_i + \frac{\varepsilon}{|\mathcal{V}|})^{-\rho} \zeta_u(\eta)^{\rho} \leq d_i^{\frac{\rho}{1-\rho}} \leq d_{min}^{\frac{\rho}{1-\rho}}$, where d_{min} is the minimum of d_v $(v \in \mathcal{V})$. Hence $P_1 \leq |\mathcal{V}| \sigma^{\bar{\rho}} \|\bar{\alpha}\|^{2\bar{\rho}} d_{min}^{\frac{\rho}{1-\rho}}$. In addition, since

$0 \leq \bar{\alpha} \leq C$, we have $\|\bar{\alpha}\| \leq \sqrt{mTC}$. Summarizing these findings, we obtain the following bound on the gradient:

$$\|\nabla g(\eta)\|_1 \leq \frac{(1-\varepsilon)}{2\bar{\rho}} mTC^2 \theta^{1-\bar{\rho}} \sigma^{\bar{\rho}} \varepsilon^{\frac{1-\bar{\rho}}{\bar{\rho}}} |\mathcal{V}|^{\frac{2}{\rho}+1} d_{min}^{\frac{\rho}{1-\rho}} d_{max}^{\frac{2-\rho}{\rho-1}}$$

Proof will be similar for the gHKL$_{\text{MT}}$ formulations in other learning settings. $\square$

A.1.7 Proof of Theorem 5.3.2

Proof. Given a triplet $(\gamma, \lambda, \alpha = [\alpha_1^\top, \ldots, \alpha_T^\top]^\top)$ (with associated primal $(\mathbf{f} = (f_1, \ldots, f_T), b, \xi)$), the duality gap (Δ) between the two variational formulations in lemma A.1.1 is given by

$$\begin{aligned}
\Delta &= \max_{\hat{\alpha}_t \in S(\mathbf{y}_t, C) \forall t} G(\gamma, \lambda, \hat{\alpha}) \; - \min_{\hat{\gamma} \in \Delta_{|\mathcal{V}|,1}} \; \min_{\hat{\lambda}_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} G(\hat{\gamma}, \hat{\lambda}, \alpha) \\
&\leq \frac{1}{2}\Omega_T(\mathbf{f})^2 + C\mathbf{1}^\top\xi \; - \min_{\hat{\gamma} \in \Delta_{|\mathcal{V}|,1}} \; \min_{\hat{\lambda}_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} G(\hat{\gamma}, \hat{\lambda}, \alpha) \\
&= \overbrace{\Omega_T(\mathbf{f})^2 + C\mathbf{1}^\top\xi - \mathbf{1}^\top\alpha}^{\text{Gap in solving with fixed } (\gamma,\lambda)} + \frac{1}{2}\overbrace{\left(\max_{\hat{\gamma} \in \Delta_{|\mathcal{V}|,1}} \; \max_{\hat{\lambda}_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} \sum_{w \in \mathcal{V}} \delta_w(\hat{\gamma}, \hat{\lambda}) \alpha^\top \mathbf{Y} H_w \mathbf{Y} \alpha - \Omega_T(\mathbf{f})^2 \right)}^{\text{Gap in solving with fixed } \alpha}
\end{aligned}$$

With this upper bound on duality gap, it is easy to see that the following condition is sufficient for the reduced solution with active set $\mathcal{W}$ having $\Delta \leq \epsilon$:

$$\max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \; \max_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} \sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) \alpha_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y} \alpha_{\mathcal{W}} \; \leq \Omega_T(\mathbf{f}_{\mathcal{W}})^2 + 2(\epsilon - \epsilon_{\mathcal{W}}) \tag{A.10}$$

where $\epsilon_{\mathcal{W}}$ is the gap[1] associated with the computation of the $\alpha_{\mathcal{W}}$. Here as well as in the rest of the proof, the subscript $\cdot_{\mathcal{W}}$ implies the value of the variable obtained when the gHKL$_{\text{MT}}$ formulation is solved with $\mathcal{V}$ restricted to active set $\mathcal{W}$. In appendix A.1.5, we proved that the L.H.S. of the above inequality is equal to the R.H.S. of (A.6), *i.e.*,

$$\max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \; \max_{\lambda_v \in \Delta_{|D(v)|,\hat{\rho}} \forall v \in \mathcal{V}} \sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) M_w = \min_{\kappa \in L} \; \max_{v \in \mathcal{V}} \; d_v^{-2} \left(\sum_{w \in D(v)} \left(\kappa_{vw}^2 M_w \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}} \tag{A.11}$$

where $M_w = \alpha_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y} \alpha_{\mathcal{W}}$.

[1]This is given by the gap associated with the $\hat{\rho}$-norm MKL solver employed in the mirror descent algorithm for solving the small problem (5.9).

Next, we obtain an upper bound of the above by substituting any $\kappa \in L$ in the R.H.S of (A.11). In particular, we employ the following: the value of κ_{vw} $v, w \in \mathcal{W}$ is taken to be that obtained by solving the small[2] problem (5.9). This is fine because $\mathcal{W} = hull(\mathcal{W})$. For $v \in \mathcal{W}^c$ and $w \in \mathcal{W}$, by definition of L and $\mathcal{W}$, we have $\kappa_{vw} = 0$. Next, κ_{vw} is set to zero $\forall\, v \in \mathcal{W}, w \in \mathcal{W}^c$. For the remaining κ_{vw}, $v \in \mathcal{W}^c$ and $w \in \mathcal{W}^c$, we use the value of κ obtained by solve (A.6) for $\rho = 1$ case, $\kappa_{vw} = d_v \left(\sum_{u \in A(v) \cap \mathcal{W}^c} d_u\right)^{-1}$ [also see Section A.5 10]. Note that the above constructed value of κ is feasible in set L. With this choice of κ substituted in the R.H.S. of (A.11), we have the following inequalities

$$
\begin{aligned}
&\max_{\gamma \in \Delta_{|\mathcal{V}|,1}} \max_{\lambda_v \in \Delta_{|D(v)|,\bar{\rho}} \forall v \in \mathcal{V}} \sum_{w \in \mathcal{V}} \delta_w(\gamma, \lambda) \alpha_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y} \alpha_{\mathcal{W}} \\
&\leq \max \left\{ \Omega_T(\mathbf{f}_{\mathcal{W}})^2, \max_{u \in \mathcal{W}^c} \left(\sum_{w \in D(u)} \left(\frac{\alpha_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y} \alpha_{\mathcal{W}}}{\left(\sum_{v \in A(w) \cap \mathcal{W}^c} d_v\right)^2} \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}} \right\} && \text{(Specific choice of } \kappa) \\
&= \max \left\{ \Omega_T(\mathbf{f}_{\mathcal{W}})^2, \max_{u \in sources(\mathcal{W}^c)} \left(\sum_{w \in D(u)} \left(\frac{\alpha_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y} \alpha_{\mathcal{W}}}{\left(\sum_{v \in A(w) \cap \mathcal{W}^c} d_v\right)^2} \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}} \right\} && (\because \mathcal{W} = hull(\mathcal{W})) \\
&\leq \max \left\{ \Omega_T(\mathbf{f}_{\mathcal{W}})^2, \max_{u \in sources(\mathcal{W}^c)} \left(\sum_{w \in D(u)} \left(\frac{\alpha_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y} \alpha_{\mathcal{W}}}{\left(\sum_{v \in A(w) \cap D(u)} d_v\right)^2} \right)^{\bar{\rho}} \right)^{\frac{1}{\bar{\rho}}} \right\} \\
&\quad (\because \textstyle\sum_{v \in A(w) \cap \mathcal{W}^c} d_v \geq \sum_{v \in A(w) \cap D(u)} d_v) \\
&\leq \max \left\{ \Omega_T(\mathbf{f}_{\mathcal{W}})^2, \max_{u \in sources(\mathcal{W}^c)} \sum_{w \in D(u)} \frac{\alpha_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y} \alpha_{\mathcal{W}}}{\left(\sum_{v \in A(w) \cap D(u)} d_v\right)^2} \right\} && (\because \|\beta\|_1 \geq \|\beta\|_{\bar{\rho}}\ \forall\ \bar{\rho} \geq 1)
\end{aligned}
$$

Employing this upper bound in (A.10) leads to the result in Theorem 5.3.2. Note that in practice, the last upper bound is not loose for REL application. This is because most of the matrices, especially near the bottom of the lattice, will be (near) zero-matrices – larger the conjunctive rule, the fewer are the examples which may satisfy it. □

[2] κ_{vw} ($\forall\, v, w \in \mathcal{W}$) obtained in this manner satisfy the constraint set L restricted to $\mathcal{W}$, *i.e.*, $L_{\mathcal{W}} = \{\kappa \in \mathbb{R}^{|\mathcal{W}| \times |\mathcal{W}|} \mid \kappa \geq 0,\ \sum_{v \in A(w)} \kappa_{vw} = 1,\ \kappa_{vw} = 0\, \forall\, v \in A(w)^c \cap \mathcal{W},\ \forall\, w \in \mathcal{W}\}$

A.1.8 Extending gHKL$_{\text{MT}}$ to General Convex Loss Functions

Here we present extension of the proposed algorithm to other settings like regression. We begin with the gHKL$_{\text{MT}}$ formulation for a general convex loss function ℓ (for example, the hinge loss, the square loss, the huber loss, *etc*.).

Equation (5.3) presented the primal gHKL$_{\text{MT}}$ with a general convex loss function ℓ. The specialized dual formulation corresponding to (5.3) is given by

$$\min_{\eta\in\Delta_{|\mathcal{V}|,1}} \max_{\alpha_t\in\mathbb{R}^m, \mathbf{1}^\top\alpha_t=0\ \forall t} -C\sum_{t=1}^{T}\sum_{i=1}^{m}\ell^*\left(-\frac{\alpha_{ti}}{C}, y_{ti}\right) - \frac{1}{2}\left(\sum_{w\in\mathcal{V}}\zeta_w(\eta)\left(\alpha^\top H_w\alpha\right)^{\bar{\rho}}\right)^{\frac{1}{\bar{\rho}}}$$

where $\alpha = [\alpha_1^\top, \ldots, \alpha_T^\top]^\top$, $\zeta_w(\eta) = \left(\sum_{v\in A(w)} d_v^\rho \eta_v^{1-\rho}\right)^{\frac{1}{1-\rho}}$ (refer Theorem 5.2.3 for details) and ℓ^* denotes the Fenchel conjugate[3] function [26] of ℓ.

A.1.9 Prediction function for gHKL$_{\text{MT}}$ with the hinge loss function

Let the final active set be $\mathcal{W}$ and $(\bar{\eta}_{\mathcal{W}}, \bar{\alpha}_{\mathcal{W}})$ be the optimal solution of (5.10). Then the prediction function for an instance $\mathbf{x}_{tj}$ belonging to the t^{th} task is given by

$$F_t(\mathbf{x}) = (\bar{\alpha}_{\mathcal{W}} \odot \mathbf{y})^\top \left(\sum_{w\in\mathcal{W}} \bar{\theta}_w(\zeta_w(\bar{\eta}_{\mathcal{W}}))^{\frac{1}{\bar{\rho}}} H_w(\cdot, \mathbf{x}_{tj})\right) \tag{A.12}$$

where symbol $\odot$ denote element-wise product, H_w is the kernel matrix corresponding to the multi-task kernel (5.8), $H_w(\cdot, \mathbf{x}_{tj}) = ((H_w(\mathbf{x}_{t'i}, \mathbf{x}_{tj}))_{i=1}^m)_{t'=1}^T$ and

$$\bar{\theta}_w = \left(\frac{(\zeta_w(\bar{\eta}_{\mathcal{W}}))^{\frac{1}{\bar{\rho}}}\bar{\alpha}_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y}\bar{\alpha}_{\mathcal{W}}}{\left(\sum_{w\in\mathcal{W}}\left((\zeta_w(\bar{\eta}_{\mathcal{W}}))^{\frac{1}{\bar{\rho}}}\bar{\alpha}_{\mathcal{W}}^\top \mathbf{Y} H_w \mathbf{Y}\bar{\alpha}_{\mathcal{W}}\right)^{\bar{\rho}}\right)^{\frac{1}{\bar{\rho}}}}\right)^{\frac{1}{\bar{\rho}-1}}$$

A.1.10 Proof of Corollary 5.3.3

Note that proving the computational complexity of the matrix $\mathcal{K}_u$ $(u \in sources(\mathcal{W}^c))$ in (5.12) to be polynomial time in size of the active set and the training set dimensions suffices to prove

[3]Fenchel conjugate $\varphi^*(z)$ of a convex function $\varphi(u)$ is given by $\varphi^*(z) = \sup_u z^\top u - \varphi(u)$. As an example, for hinge loss $\ell(u, y) = \max(0, 1 - uy)$, $\ell^*(z, y) = \begin{cases} zy & \text{if } zy \in [-1, 0] \\ \infty & \text{otherwise} \end{cases}$

the corollary. This is because all the other steps in Algorithms 3 and 2 are of polynomial time complexity (discussed in Section 5.3).

We begin the proof by introducing some indexing notations related to the multi-task matrices. Let the entries in H_w, the $mT \times mT$ multi-task kernel matrix, be arranged in the following form: the entry corresponding to the input pair $(\mathbf{x}_{t_1 i}, \mathbf{x}_{t_2 j})$ be in the $((t_1 - 1) * m + i)^{th}$ row and $((t_2 - 1) * m + j)^{th}$ column of H_w.

Next we observe that the expression for $\mathcal{K}_u$ in Theorem 5.3.2 may be rewritten as

$$\mathcal{K}_u = \underbrace{\left(\sum_{w \in D(u)} \frac{K_w}{\left(\sum_{v \in A(w) \cap D(u)} d_v \right)^2} \right)}_{T_u} \odot K_T$$

where: i) K_w is a $mT \times mT$ matrix corresponding to the base kernel k_w and constructed from the inputs from all the tasks, ii) K_T is a $mT \times mT$ such that the entry corresponding to the $((t_1 - 1) * m + i)^{th}$ row and $((t_2 - 1) * m + j)^{th}$ column $(1 \leq i, j \leq m)$ of K_T is $B(t_1, t_2)$, and iii) $\odot$ is the symbol for element-wise product (Hadamard product).

In the above expression, $\mathcal{K}_u$ is computable in polynomial time if and only if T_u is computable in polynomial time. The proof of the corollary follows from observing the expression of the sufficiency condition for optimality of the HKL (refer to Equation 21 in [10]), which also involves the term T_u.

A.1.11 Proof of Theorem 5.4.1

Given an active set $\mathcal{W}$ of size W, proving that the computational complexity of the verification of the sufficient condition of optimality (5.12) is polynomial in terms of the active set and the training set sizes suffices to prove Theorem 5.4.1. This is because all the other steps in Algorithms 3 and 2 are of polynomial time complexity (Section 5.3).

In the REL setup, the DAG is the conjunction lattice and the embedded kernels k_v $v \in \mathcal{V}$ can be rewritten as

$$k_v(\mathbf{x}_i, \mathbf{x}_j) = \phi_v(\mathbf{x}_i) \cdot \phi_v(\mathbf{x}_j) = \left(\prod_{c \in S_v} \phi_c(\mathbf{x}_i) \right) \cdot \left(\prod_{c \in S_v} \phi_c(\mathbf{x}_j) \right) = \bigodot_{c \in S_v} k_c(\mathbf{x}_i, \mathbf{x}_j)$$

where S_v is the set of basic propositions involved in the conjunction ϕ_v and $\odot$ is the symbol for element-wise product (Hadamard product). The kernels corresponding to the basic propositions are in fact the base kernels embedded in the second level nodes of the lattice $\mathcal{V}$. Employing the above, the matrices $\mathcal{K}_u$ (in L.H.S. of (5.12)) can be computed as

$$\mathcal{K}_u = \sum_{w \in D(u)} \frac{K_w}{\left(\sum_{v \in A(w) \cap D(u)} d_v\right)^2} = \left(\bigodot_{c \in S_u} \frac{K_c}{a^2}\right) \odot \left(\bigodot_{c \in B/S_u} \left(\frac{K_c}{(1+a)^2} + \mathbf{1}\mathbf{1}^\top\right)\right)$$

where K_c is the kernel matrix corresponding to the basic proposition ϕ_c, B is the set of all basic propositions and the parameters d_v $(v \in \mathcal{V})$ are defined as $d_v = a^{|S_v|}$ $(a > 0)$.

It is obvious that $\mathcal{K}_u$ $(u \in \mathcal{V})$ can be computed in $O(pm^2)$. In practice, by caching the matrix $\mathcal{K}_u$ associated with a node u, the corresponding matrix $\mathcal{K}_w$ associated with any child w of u can be computed in $O(m^2)$. For illustration, suppose $\mathcal{K}_{u_1}$ needs to be computed, given that $\mathcal{K}_{u_0}$ is cached and u_0 is a parent of u_1. Let the extra basic proposition contained in ϕ_{u_1} (with respect to ϕ_{u_0}) be ϕ_e. Then $\mathcal{K}_{u_1}$ can be calculated as follows:

$$\mathcal{K}_{u_1} = \mathcal{K}_{u_0} \odot \left(\frac{K_e}{a^2}\right) \oslash \left(\frac{K_e}{(1+a)^2} + \mathbf{1}\mathbf{1}^\top\right)$$

where $\oslash$ is the symbol for element-wise division of matrices.

Hence, plugging the REL specific values in the runtime complexity of the gHKL algorithm, ω =constant and $z = p$, the runtime complexity of the gHKL based REL algorithm is $O(m^3 W^3 \log(W) + m^2 W^2 p)$.

A.1.12 REL Binary Classification Results in AUC

Dataset	RuleFit	SLI	ENDER	HKL-ℓ_1-MKL	gHKL$_\rho$		
					$\rho = 2$	$\rho = 1.5$	$\rho = 1.1$
TIC-TAC-TOE	0.736 ± 0.05	0.482 ± 0.21	0.783 ± 0.036	0.836 ± 0.024	0.967 ± 0.023	0.973 ± 0.02	$\mathbf{0.975 \pm 0.018}$
B-CANCER-W	0.941 ± 0.011	0.917 ± 0.051	0.958 ± 0.039	0.981 ± 0.008	$\mathbf{0.984 \pm 0.005}$	0.93 ± 0.099	0.93 ± 0.099
DIABETES	0.67 ± 0.027	0.576 ± 0.115	0.761 ± 0.02	0.746 ± 0.050	$\mathbf{0.766 \pm 0.046}$	0.733 ± 0.058	0.636 ± 0.118
HABERMAN	0.537 ± 0.054	0.17 ± 0.155	$\mathbf{0.575 \pm 0.039}$	0.524 ± 0.078	0.556 ± 0.07	0.482 ± 0.11	0.383 ± 0.166
HEARTC	0.764 ± 0.03	0.541 ± 0.215	0.805 ± 0.031	0.802 ± 0.085	$\mathbf{0.837 \pm 0.035}$	0.763 ± 0.12	0.753 ± 0.118
BLOOD TRANS.	0.546 ± 0.06	0.175 ± 0.256	$\mathbf{0.68 \pm 0.028}$	0.660 ± 0.025	0.667 ± 0.034	0.634 ± 0.028	0.519 ± 0.079
HEARTSTAT	0.765 ± 0.028	0.712 ± 0.085	0.801 ± 0.022	0.825 ± 0.032	$\mathbf{0.849 \pm 0.021}$	0.83 ± 0.027	0.811 ± 0.056
MONK-3	0.972	0.632	0.998	0.995	**1**	0.998	0.957
VOTE	0.955 ± 0.022	0.919 ± 0.048	0.965 ± 0.014	$\mathbf{0.977 \pm 0.009}$	0.972 ± 0.016	0.948 ± 0.015	0.945 ± 0.016
B-CANCER	0.578 ± 0.05	0.469 ± 0.078	0.622 ± 0.043	0.627 ± 0.063	$\mathbf{0.637 \pm 0.055}$	0.576 ± 0.089	0.513 ± 0.124
MAM. MASS	0.818 ± 0.02	0.763 ± 0.08	$\mathbf{0.887 \pm 0.006}$	0.866 ± 0.028	0.882 ± 0.023	0.85 ± 0.032	0.839 ± 0.03
LIVER	0.607 ± 0.017	0.093 ± 0.168	0.619 ± 0.038	0.619 ± 0.074	$\mathbf{0.623 \pm 0.038}$	0.583 ± 0.11	0.565 ± 0.109

Table A.1: Results on binary REL classification. AUC along with standard deviation averaged over 10 splits is shown.

A.2 Learning Kernels on Latent Task Structure

A.2.1 Proof of Theorem 6.3.1

In the following, we will first reparameterize the proposed primal formulation (6.1) with the regularizer defined in (6.3). We re-write the primal formulation below

$$\min_{\mathbf{h},\mathbf{b}} \frac{1}{2}\left[\sum_{v\in\mathcal{V}} d_v \left\{ \sum_{w\in D(v)} \left(\sum_{j=1}^{n} \left(\mu\|h^j_{w0}\|_2^2 + \sum_{t\in w}\|h^j_{tw}\|_2^2\right)^{\frac{p}{2}}\right)^{\frac{1}{p}}\right\}^{\frac{1}{q}}\right]^2 + C\sum_{t,i}\ell(F_t(\mathbf{x}_{ti}), y_{ti})$$

Without loss of generality, we assume that within each node w, the tasks are arranged in the order of their number (hence t_i before t_j if $t_i < t_j$). Next, we employ the following multi-task feature map [48]. For a given input instance $\mathbf{x}$ belonging to the task t, its feature map with respect to the kernel k^j and a group w (containing task t) is: $\Phi^j_w(\mathbf{x}, t) =$

$(\frac{\phi^j(\mathbf{x})}{\sqrt{\mu}}, \underbrace{\mathbf{0},\ldots,\mathbf{0}}_{\text{tasks before t}}, \phi^j(\mathbf{x}), \underbrace{\mathbf{0},\ldots,\mathbf{0}}_{\text{tasks after t}})$, where $\mathbf{0}$ represents a zero vector in the induced feature space ϕ^j of the kernel k^j. The task parameters within the group corresponding to kernel k^j are re-grouped as: $f_w^j = (\sqrt{\mu}h_{0w}^j, h_{t_1 w}^j, \ldots, h_{t_{|w|}w}^j)$. Hence, the decision function of task t is re-written as: $F_t(\mathbf{x}) = \sum_{w\in\mathcal{G}_t}\langle f_w, \Phi_w(\mathbf{x}, t)\rangle - b_t$, where $\Phi_w = (\Phi_w^1, \ldots, \Phi_w^n)$ and $f_w = (f_w^1, \ldots, f_w^n)$. From such constructions, it is evident that the the mixed-norm base regularizer $\|\Theta_w\|_p$ within each group w can be written as: $\|\Theta_w\|_p = \left(\sum_j \|f_w^j\|_2^p\right)^{\frac{1}{p}} = \|f_w\|_p$. Using the above reparameterization, the proposed primal formulation with hinge loss function can be equivalently stated as:

$$\min_{\mathbf{f},\xi,\mathbf{b}} \frac{1}{2}\Omega(\mathbf{f})^2 + C\sum_{t=1}^{T}\sum_{i=1}^{m}\xi_{ti} \tag{A.13}$$
$$\text{s.t. } y_{ti}F_t(\mathbf{x}_{ti}) \geq 1 - \xi_{ti},\ \xi_{ti} \geq 0\ \forall\, i, t$$

where $\Omega(\mathbf{f}) = \sum_{v\in\mathcal{V}} d_v\|f_{D(v)}\|_{q,p}$ and $\|f_{D(v)}\|_{q,p} = \left(\sum_{w\in D(v)}\|f_w\|_p^q\right)^{\frac{1}{q}}$.

Note that (A.13) aims at finding an optimal vector in an RKHS. In order to facilitate the application of a suitable representer theorem [113] for writing down the dual, we employ a variational formulation of $\Omega(\mathbf{f})$. By repeated application of lemma (appendix A.1.1), we get the following result (see also A.1.2):

$$\Omega(\mathbf{f})^2 = \min_{\gamma\in\Delta_{|\mathcal{V}|,1}}\ \min_{\lambda_v\in\Delta_{|D(v)|,\hat{q}}\forall v\in V}\ \min_{\sigma_v\in\Delta_{k,\hat{p}}\forall v\in V}\sum_{w\in\mathcal{V}}\delta_w^{-1}(\gamma,\lambda)\frac{\|f_w\|_2^2}{\sigma_w^j}$$

where $\delta_w^{-1}(\gamma,\lambda) = \sum_{v\in A(w)}\frac{d_v^2}{\gamma_v\lambda_{vw}}$. Next, by applying the representer theorem (see also [108]) and using the notion of dual norm [26], we derive a partial-dual (dual with respect to variables $\mathbf{f}, b, \xi$ alone) of (A.13) to be the following:

$$\min_{\gamma\in\Delta_{|\mathcal{V}|,1}}\ \min_{\lambda_v\in\Delta_{|D(v)|,\hat{q}}\forall v\in\mathcal{V}}\ \max_{\alpha_t\in S_m(\mathbf{y}_t,C)\forall t\in\mathcal{T}} G(\gamma,\lambda,\alpha) \tag{A.14}$$

where

$$G(\gamma,\lambda,\alpha) = \sum_{t,i}\alpha_{ti} - \frac{1}{2}\sum_{w\in\mathcal{V}}\delta_w(\gamma,\lambda)\|\alpha^\top\mathbf{K}_w\alpha\|_{\bar{p}},$$

$\|\alpha^\top\mathbf{K}_w\alpha\|_{\bar{p}} = \left(\sum_{j=1}^n\left(\alpha^\top\mathbf{K}_w^j\alpha\right)^{\bar{p}}\right)^{\frac{1}{\bar{p}}}$ and $\bar{p} = \frac{p}{2(p-1)}$.

We obtain the highly specialized partial dual 6.4 from the Lagrangian partial dual A.14 via a series of optimization related reformulations (see A.1.4 and A.1.5 for details).

A.2.2 Proof of Theorem 6.4.1

The proof of this theorem follows very closely the proof technique employed for Theorem 5.3.2. The latter is detailed in Section A.1.7. The key idea is to upper bound the duality gap associated with the min-max interchange within the Lagrangian partial dual A.14.

A.3 Kernel Selection Path in ℓ_p-norm Multiple Kernel Learning

A.3.1 Proof of Theorem 7.3.1

Proof. The optimal solution path of (1), i.e. $\eta^*(p)$, is unique since the objective of (1) is strictly convex. In order to prove the smoothness of the optimal solution path, we begin by deriving the necessary and sufficient conditions of optimality for (1). To this end, we first define

$$g(\eta_i) = \lambda_1 \bar{B}_F(\eta_i) + \lambda_2 \eta_i^p - \eta_i \mathbf{y}^\top K^i \mathbf{y}$$

Next, we consider two cases:

Case 1: $\eta_i^* = 0$ for a given p: The necessary and sufficient conditions of optimality for a convex function is that the gradient of the function at optimality, $g'(\eta_i^*)$, should lie in the normal cone of the feasibility set [22]. This results in the following inequality:

$$\lambda_1(F'(0) - F'(\eta_i^0)) - \mathbf{y}^\top K^i \mathbf{y} \geq 0 \tag{A.15}$$

Since F is a convex function, we have $F(\eta_i) \geq F(\eta_i^0) + F'(\eta_i^0)(\eta_i - \eta_i^0)$ as well as $F(\eta_i^0) \geq F(\eta_i) + F'(\eta_i)(\eta_i^0 - \eta_i)$. Summing these two inequalities, we get $0 \geq (F'(\eta_i) - F'(\eta_i^0))(\eta_i^0 - \eta_i)$. Now substituting $\eta_i = 0$, we get $0 \geq \eta_i^0(F'(0) - F'(\eta_i^0))$. Since $\eta_i^0 \geq 0$ (feasible set of η_i is ≥ 0), it follows $0 \geq F'(0) - F'(\eta_i^0)$. Hence, in the LHS of (A.15), $(F'(0) - F'(\eta_i^0)) \leq 0$ and $-\mathbf{y}^\top K^i \mathbf{y} \leq 0$. It follows that the optimality conditions for $\eta_i^* = 0$ are

$$F'(0) - F'(\eta_i^0) = 0 \text{ and } \mathbf{y}^\top K^i \mathbf{y} = 0 \tag{A.16}$$

Note that both the above equalities are independent of p. It follows that if $\exists\, p' > 1\ s.t.\ \eta_i^*(p') = 0$ then $\eta_i^*(p) = 0\ \forall\, p > 1$. Thus, the optimal solution path, $\eta_i^*(p)$, is smooth, in fact linear, when $\exists\, p' > 1\ s.t.\ \eta_i^*(p') = 0$.

Case 2: $\eta_i^* > 0$ for a given p: In this case, the necessary and sufficient conditions of optimality [22] simplifies to $g'(\eta_i^*) = 0$. Thus we get the following optimality condition

$$\begin{aligned}\mathbf{G}_i(\eta_i, p) \equiv & \lambda_1(F'(\eta_i) - F'(\eta_i^0)) + \lambda_2 p \eta_i^{p-1} \\ & - \mathbf{y}^\top K^i \mathbf{y} = 0\end{aligned} \tag{A.17}$$

In the following, we prove that the path of the optimal solution, $\eta_i^*(p)$, of (A.17) is smooth for the non trivial case: $\eta_i^*(p) > 0 \, \forall \, p > 1$.

Since the pair $(\eta_i^*(p), p)$ always satisfy the equality in (A.17), we must have that $\mathrm{d}\mathbf{G}_i(\eta_i^*(p), p) \equiv \sum_j \frac{\partial \mathbf{G}_i}{\partial \eta_j} \mathrm{d}\eta_j + \frac{\partial \mathbf{G}_i}{\partial p} \mathrm{d}p = 0 \, \forall \, p > 1$. This leads to

$$\frac{\mathrm{d}\eta_i^*(p)}{\mathrm{d}p} = \frac{-\lambda_2 \eta_i^*(p)^{p-1}(1 + p \ln \eta_i^*(p))}{\lambda_1 F''(\eta_i^*(p)) + \lambda_2 p(p-1)\eta_i^*(p)^{p-2}} \tag{A.18}$$

The terms $\ln \eta_i^*(p)$ and $\eta_i^*(p)^{p-2}$ are always finite as $\eta_i^*(p) > 0$ and the denominator of (A.18) is always non-zero, in fact positive, because for any convex, twice-differentiable F we have: $F''(\eta_i^*(p)) \geq 0$. Hence, the derivative along the optimal solution path (A.18) is well defined and itself a continuous function; proving that the optimal solution path is smooth (but generally non-linear) in this case too. □

A.3.2 Proof of Theorem 7.3.2

Proof. The monotonic behavior of $\eta_i^*(p)$ follows from observing the sign of (A.18) and from the fact that $e^{-\frac{1}{p}}$ is a monotonically increasing function of p. Note that the denominator of (A.18) is positive. □

A.3.3 Analysis of ℓ_p-MKL with Square Loss

In this section, we analyze the proposed conjecture in the context of the ℓ_p-MKL formulation for the ridge regression [39]:

$$\min_{\eta \geq 0} \max_{\alpha \in \mathbb{R}^m} \mathbf{y}^\top \alpha - \frac{1}{2} \alpha^\top Q_\eta \alpha + \lambda_2 \sum_{i=1}^{n} \eta_i^p \tag{A.19}$$

where $Q_\eta = \sum_i \eta_i K^i + \frac{I}{2\lambda_1}$, I is the $m \times m$ identity matrix and $\lambda_1, \lambda_2 > 0$ are the regularization parameters. Firstly, we consider a setting employed in [83, 44, 43], which leads to the selection of a low-dimensional subspace: unit rank base kernels $K^i = \mathbf{u}_i\mathbf{u}_i^\top$ s.t. $\text{trace}(K^i) = 1$ and $\langle K^i, K^j\rangle = 0\,\forall\, i \neq j$. The following theorem holds in this setting:

Theorem A.3.1. *Let $\eta_i^*(p)$ denote the optimal weight corresponding to the i-th kernel in (A.19) at p. Given $\eta_i^*(p')$, the following holds as p decreases from p' to unity: (1) $\eta_i^*(p)$ decreases monotonically whenever $\eta_i^*(p') < e^{-1}$; (2) $\eta_i^*(p)$ increases monotonically whenever $\eta_i^*(p') > e^{-\frac{1}{p'}}$*

The proof of the above theorem involves deriving and analyzing the $\mathrm{d}\eta^*(p)/\mathrm{d}p$ term.Needless to say, the above theorem implies that the conjecture is true for the case mentioned above. We now present an interesting example where the conjecture does not hold with $\epsilon = 0$:

Theorem A.3.2. *Consider the regression formulation in (A.19) with two given base kernels of unit rank and unit trace: k^1 and k^2. Additionally, let $\mathbf{y}^\top K^2K^1\mathbf{y} > \mathbf{y}^\top K^1\mathbf{y} > 0$. Then for some λ_1, λ_2, $\exists\, p' > 1$ such that $\eta_1 = 0$ if and only if $p = p'$.*

Proof. The KKT conditions for optimality for the convex optimization problem (A.19) are:

$$\mathbf{G}_i(\eta, p) \equiv -\frac{1}{2}\mathbf{y}^\top Q_\eta^{-1}K^iQ_\eta^{-1}\mathbf{y} + \lambda_2 p\eta_i^{p-1} = 0 \tag{A.20}$$

where $Q_\eta = \sum_i \eta_i K^i + \frac{I}{2\lambda_1}$ and $i = 1, \ldots, r$. In the following, we first obtain a particular $p'(> 1)$ where $\eta_1^*(p') = 0$. Next, from (A.20), we show that η_1^* is zero only at $p = p'$.

Let $\eta_1^* = 0$ at $p = p'$. From the KKT conditions (A.20) corresponding to $\mathbf{G}_1(\eta, p)$, we get $\mathbf{y}^\top Q_\eta^{-1}K^1Q_\eta^{-1}\mathbf{y} = 0$ (as $\eta_1^* = 0$). Employing the Sherman-Morrison formula for computing Q_η^{-1} and simplifying the L.H.S. of the above expression yields

$$\eta_2^* = \frac{\mathbf{y}^\top K^1\mathbf{y}}{2\lambda_1(\mathbf{y}^\top K^2K^1\mathbf{y} - \mathbf{y}^\top K^1\mathbf{y})} \tag{A.21}$$

Since we have $\mathbf{y}^\top K^2K^1\mathbf{y} > \mathbf{y}^\top K^1\mathbf{y} > 0$, it follows $\eta_2^* > 0$. Similarly, from the KKT conditions (A.20) corresponding to $\mathbf{G}_2(\eta, p)$, we obtain

$$p'(\eta_2^*)^{p'+1} = \frac{1}{2\lambda_2}\left(\frac{\mathbf{y}^\top K^1\mathbf{y}\mathbf{y}^\top K^2\mathbf{y}}{\mathbf{y}^\top K^2K^1K^2\mathbf{y}}\right) \tag{A.22}$$

Note that $\mathbf{y}^\top K^2K^1\mathbf{y} > 0 \Rightarrow \mathbf{y}^\top K^2K^1K^2\mathbf{y} > 0$. Now, consider the following values of parameters (λ_1, λ_2):

$$\lambda_1 = \frac{\mathbf{y}^\top K^1\mathbf{y}}{2(\mathbf{y}^\top K^2K^1\mathbf{y} - \mathbf{y}^\top K^1\mathbf{y})},\ \ \lambda_2 = \frac{\mathbf{y}^\top K^1\mathbf{y}\mathbf{y}^\top K^2\mathbf{y}}{3\mathbf{y}^\top K^2K^1K^2\mathbf{y}}$$

Note that both $\lambda_1, \lambda_2 > 0$ in the above case. Employing these in (A.21) and (A.22), we obtain the optimal kernel weight (η_1^*, η_2^*) at $p' = 1.5$ as $(0, 1)$. Moreover, with the above mentioned values of parameters (λ_1, λ_2), it also follows that at optimality, $(\eta_1^* = 0) \Rightarrow (p' = 1.5)$ and $(\eta_1^* = 0) \Rightarrow (\eta_2^* = 1)$. Hence, it follows that $\eta_1^* > 0$ at any $1 < p < p'$. $\square$

This theorem shows a case in which the kernel weight, after attaining the lowest feasible value (zero), at $p = p'$, grows as p further decreases from p'. Figure A.1 shows an instance from a real world data set (Parkinson disease data set from the UCI Repository) where the optimal kernel weight in (A.19) grows after attaining zero. We can observe that at around $p = 1.5$, the optimal kernel weight corresponding to Feature 1 is zero, and it again starts growing as p is further decreased. This observation show that the conjecture is not universally true for low values of ϵ. However, in the same setting as in Theorem A.3.2, the conjecture may be true for some other $\epsilon > 0$. The proposed algorithm is usable whenever such an ϵ is small[4].

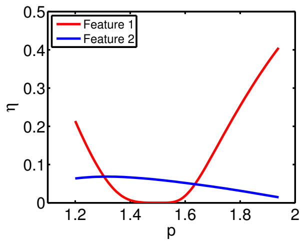

Figure A.1: η_1^* decrease as p is decreased from 2, attains zero at around $p = 1.5$ and again grows as p decreases further. This example proves that the monotonicity conjecture does not universally hold for (A.19) when $\epsilon = 0$.

In summary, the proposed conjecture is itself non-trivial and requires careful analysis for different ℓ_p-MKL formulations. Interestingly, in case of the proposed Generalized ℓ_p-KTA, it holds for small enough tolerance, $\epsilon < e^{-1}$.

A.3.4 Second order derivative along the solution path

Here, we derive the second order derivative along the optimal solution path of (7.1), which can be employed in Algorithm 5, assuming that the function F is thrice differentiable. From (A.18),

[4]Interestingly, in our initial simulations on benchmark data sets, we found that many existing l_p-MKL formulations do satisfy the conjecture at a reasonably low $\epsilon > 0$. We postpone theoretical analysis of this very interesting and open question to future.

we have the following form: $\frac{\mathbf{d}\eta_i^*(p)}{\mathbf{d}p} = f(\eta_i^*(p), p)$ where f is a function corresponding to the R.H.S. of (A.18). Hence, employing the total derivative formula in the above equation, we get the following formulation for the second order derivative:

$$\frac{\mathbf{d}^2\eta_i^*(p)}{\mathbf{d}p^2} = -\frac{\lambda_2}{D}\left[\frac{\lambda_1}{\lambda_2}F'''(\eta_i^*(p)) + p(p-1)(p-2)\eta_i^*(p)^{p-3}\left(\frac{\mathbf{d}\eta_i^*(p)}{\mathbf{d}p}\right)^2\right.$$
$$\left. + 2\eta_i^*(p)^{p-2}\left(2p-1+p^2\ln(\eta_i^*(p)) - p\ln(\eta_i^*(p))\right)\frac{\mathbf{d}\eta_i^*(p)}{\mathbf{d}p} + \eta_i^*(p)^{p-1}\ln(\eta_i^*(p))\left(2+p\ln(\eta_i^*(p))\right)\right]$$

where $D = \lambda_1 F'''(\eta_i^*(p)) + \lambda_2 p(p-1)\eta_i^*(p)^{p-2}$ and the term $\mathbf{d}\eta_i^*(p)/\mathbf{d}p$ can be obtained from (A.18).

A.3.5 Lemma for generalized KL-divergence

In the case of generalized KL-divergence as the Bregman divergence in (7.1), using the optimality conditions for the non-trivial case (A.17) and Theorem 7.3.2, the following lemma is immediate:

Lemma A.3.3. *For any p, the deviation in the objective value of (7.1) obtained using the approximate path following algorithm from the true optimal objective is upper bounded by $n(\lambda_1\epsilon + \lambda_2(p-1)\epsilon^p)$.*

A.3.6 Experimental Results

Table A.2 reports the mean accuracies and standard deviations corresponding to Table 7.2 in Chapter 7.

Table A.2: The maximum classification accuracy achieved (mean and standard deviations) along the feature selection path. Generalized ℓ_p-KTA achieves significantly higher accuracies as compared to state-of-the-art KTA and $\ell_{p\geq 1}$-MKL formulations as well as leading feature selection techniques such as BAHSIC. The table reports mean and standard deviations results averaged over 5-fold cross validation. '-' denote results where the data set was too large for the feature selection algorithm to generate results.

	Arcene	Madelon	Relathe	Pcmac	Basehock	Dorothea
Gen l_p-KTA	$\mathbf{92.00} \pm 5.70$	$\mathbf{65.70} \pm 0.99$	$\mathbf{92.57} \pm 0.30$	$\mathbf{93.62} \pm 1.67$	$\mathbf{98.59} \pm 0.58$	$\mathbf{94.75} \pm 1.30$
Centered-KTA	75.00 ± 9.35	62.45 ± 2.07	90.40 ± 0.21	93.05 ± 1.44	97.29 ± 1.05	-
SMO-MKL	82.00 ± 5.70	62.05 ± 0.54	-	-	-	-
BAHSIC	69.00 ± 6.52	53.90 ± 2.97	85.07 ± 1.42	89.55 ± 0.22	93.58 ± 1.38	90.63 ± 0.77
PF-l_1-MKL	81.00 ± 6.52	62.76 ± 2.40	85.67 ± 1.90	-	-	-
PF-l_1-SVM	77.00 ± 12.04	61.25 ± 1.08	89.00 ± 2.16	90.68 ± 0.59	97.24 ± 0.89	93.88 ± 1.65
Uniform	81.00 ± 6.52	59.85 ± 0.84	90.96 ± 0.77	92.49 ± 0.64	97.99 ± 0.59	91.38 ± 1.42

Table A.3 reports the mean accuracies and standard deviations corresponding to Table 7.3 in Chapter 7.

Table A.3: The maximum classification accuracy achieved (mean and standard deviations) along the feature selection path on the ASU data sets. In keeping with the ASU experimental protocol, all algorithms are restricted to selecting at most 200 features and are allowed to train on only half the data. Generalized ℓ_p-KTA (RBF) outperforms all the linear techniques and this demonstrates the advantages of non-linear feature selection. Amongst the linear methods, our proposed method with linear features is the best in general.

	Arcene	Madelon	Relathe	Pcmac	Basehock	Dorothea
Gen l_p-KTA (RBF)	76.80 ± 9.25	64.50 ± 1.14	89.40 ± 0.93	89.76 ± 0.87	95.46 ± 1.02	93.75 ± 1.18
Gen l_p-KTA (Linear)	$\mathbf{73.40} \pm 7.06$	62.04 ± 0.97	$\mathbf{88.39} \pm 0.87$	88.88 ± 2.61	94.76 ± 0.96	$\mathbf{93.60} \pm 1.30$
Inf. Gain	72.00 ± 5.89	61.63 ± 0.95	84.39 ± 0.94	$\mathbf{88.99} \pm 1.22$	95.26 ± 1.29	93.33 ± 0.97
Chi-Square	71.20 ± 7.90	61.69 ± 1.28	83.48 ± 0.80	88.24 ± 1.39	$\mathbf{95.28} \pm 1.26$	93.33 ± 1.34
Fisher Score	66.20 ± 9.68	61.47 ± 1.04	83.35 ± 1.05	88.02 ± 1.60	94.61 ± 1.47	93.30 ± 1.32
mRMR	68.20 ± 7.33	61.87 ± 1.17	75.01 ± 1.01	83.34 ± 1.18	88.88 ± 1.00	93.18 ± 1.35
ReliefF	68.40 ± 7.71	$\mathbf{62.06} \pm 1.21$	77.08 ± 2.73	80.76 ± 1.79	86.05 ± 3.55	93.33 ± 0.93
Spectrum	64.00 ± 6.60	60.19 ± 0.76	69.99 ± 1.90	66.74 ± 1.52	69.79 ± 1.31	90.28 ± 1.25
Gini Index	64.60 ± 5.50	59.43 ± 2.38	69.50 ± 2.09	66.60 ± 1.37	69.49 ± 1.35	90.28 ± 1.25
K.-Wallis	60.20 ± 10.26	55.04 ± 1.23	70.97 ± 2.02	65.20 ± 1.31	70.37 ± 1.05	90.08 ± 1.07

Bibliography

[1] J. Aflalo, A. Ben-Tal, C. Bhattacharyya, J. Saketha Nath, and S. Raman. Variable sparsity kernel learning. *Journal of Machine Learning Research*, 12:565–592, 2011.

[2] E. L. Allgower and K. Georg. Continuation and path following. *Acta Numer*, 2:1–64, 1993.

[3] M. A. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: a review. *Foundations and Trends in Machine Learning*, 4:195–266, 2012.

[4] Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the International Conference on Machine Learning*, 2007.

[5] R. K. Ando and T. Zhang. A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

[6] G. Andrew and J. Gao. Scalable training of L_1-regularized log-linear models. In *Proceedings of the International Conference on Machine Learning*, pages 33–40, 2007.

[7] C. Archambeau, S. Guo, and O. Zoeter. Sparse bayesian multi-task learning. In *Advances in Neural Information Processing Systems*, 2011.

[8] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73:243–272, 2008.

[9] A. Argyriou, C. A. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. In *Advances in Neural Information Processing Systems*, 2007.

[10] F. Bach. High-dimensional non-linear variable selection through hierarchical kernel learning. Technical report, INRIA, France, 2009.

[11] F. R. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *Advances in Neural Information Processing Systems*, pages 105–112, 2008.

[12] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the International Conference on Machine Learning*, 2004.

[13] F. R. Bach, R. Thibaux, and M. I. Jordan. Computing regularization paths for learning multiple kernels. In *Advances in Neural Information Processing Systems*, 2004.

[14] B. Bakker and T. Heskes. Task clustering and gating for bayesian multi-task learning. *Journal of Machine Learning Research*, 4:83–99, 2003.

[15] S. Baluja and H. Rowley. Boosting sex identification performance. *International Journal of Computer Vision*, 71(1):111–119, 2007.

[16] Arindam Banerjee, Srujana Merugu, Inderjit S. Dhillon, and Joydeep Ghosh. Clustering with bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, December 2005.

[17] J. J. Baxter. A bayesian/information theoretic model of learning to learn viamultiple task sampling. *Machine Learning*, 28:7–39, 1997.

[18] J. J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[19] A. Beck and M. Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31:167–175, 2003.

[20] S. Ben-David and R. Schuller. A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine Learning*, 73:273–287, December 2008.

[21] A. Ben-Tal, T. Margalit, and A. Nemirovski. The ordered subsets mirror descent optimization method with applications to tomography. *SIAM Journal of Opimization*, 12(1):79–108, July 2001.

[22] A. Ben-Tal and A. Nemirovski. *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*. Society for Industrial and Applied Mathematics, 2001.

[23] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[24] S. Bickel, J. Bogojeska, T. Lengauer, and T. Scheffer. Multi-task Learning for HIV Therapy Screening. In *Proceedings of the International Conference on Machine Learning*, 2008.

[25] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, March 2002.

[26] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[27] B. Cao, D. Shen, J. T. Sun, Q. Yang, and Z. Chen. Feature selection in a kernel space. In *Proceedings of the International Conference on Machine Learning*, 2007.

[28] A. Caponnetto, C. A. Micchelli, M. Pontil, and Y. Ying. Universal multi-task kernels. *Journal of Machine Learning Research*, 9:1615–1646, June 2008.

[29] R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

[30] A. B. Chan, N. Vasconcelos, and G. Lanckriet. Direct convex relaxations of sparse SVM. In *Proceedings of the International Conference on Machine Learning*, pages 145–153, 2007.

[31] O. Chapelle, P. Shivaswamy, S. Vadrevu, K. Weinberger, Y. Zhang, and B. Tseng. Multitask learning for boosting with application to web search ranking. In *Proceedings of the ACM Special Interest Group on Knowledge Discovery and Data Mining*, 2010.

[32] J. Chen, S. Ji, B. Ceran, Q. Li, M. Wu, and J. Ye. Learning subspace kernels for classification. In *Proceedings of the ACM Special Interest Group on Knowledge Discovery and Data Mining*, 2008.

[33] J. Chen, L. Tang, J. Liu, and J. Ye. A convex formulation for learning shared structures from multiple tasks. In *Proceedings of the International Conference on Machine Learning*, 2009.

[34] X. Chen, Q. Lin, S. Kim, J. G. Carbonell, and E. P. Xing. Smoothing proximal gradient method for general structured sparse learning. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2011.

[35] P. Clark and T. Niblett. The CN2 Induction Algorithm. *Machine Learning*, 3:261–283, 1989.

[36] W. W. Cohen and Y. Singer. A Simple, Fast, and Effective Rule Learner. In *AAAI Conference on Artificial Intelligence*, 1999.

[37] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the International Conference on Machine Learning*, 2008.

[38] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms (3. ed.)*. MIT Press, 2009.

[39] C. Cortes, M. Mohri, and A. Rostamizadeh. L2 regularization for learning kernels. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2009.

[40] C. Cortes, M. Mohri, and A. Rostamizadeh. Learning non-linear combinations of kernels. In *Advances in Neural Information Processing Systems*, 2009.

[41] C. Cortes, M. Mohri, and A. Rostamizadeh. Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research*, 13:795–828, March 2012.

[42] C. Cortes, M. Mohri, and A. Rostamizadeh. Multi-class classification with maximum margin multiple kernel. In *Proceedings of the International Conference on Machine Learning*, 2013.

[43] N. Cristianini, H. Lodhi, and J. Shawe-taylor. Latent semantic kernels for feature selection. Technical Report NC-TR-00-080, 2000.

[44] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola. On kernel-target alignment. In *Advances in Neural Information Processing Systems*, 2001.

[45] K. Dembczyński, W. Kotłowski, and R. Słowiński. Maximum likelihood rule ensembles. In *Proceedings of the International Conference on Machine Learning*, 2008.

[46] K. Dembczyński, W. Kotłowski, and R. Słowiński. ENDER - A Statistical Framework for Boosting Decision Rules. *Data Mining and Knowledge Discovery*, 21:52–90, 2010.

[47] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

[48] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *Proceedings of the ACM Special Interest Group on Knowledge Discovery and Data Mining*, 2004.

[49] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, April 2006.

[50] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[51] J. H. Friedman and B. E. Popescu. Predictive Learning via Rule Ensembles. *Annals of Applied Statistics*, 2:916–954, 2008.

[52] G. Fung and O. L. Mangasarian. A feature selection newton method for support vector machine classification. Technical Report 02-03, Univ. of Wisconsin, 2002.

[53] N. Goernitz, C. Widmer, G. Zeller, A. Kahles, S. Sonnenburg, and G. Ratsch. Hierarchical multitask structured output learning for large-scale sequence segmentation. In *Advances in Neural Information Processing Systems*, 2011.

[54] Y. Grandvalet and S. Canu. Adaptive scaling for feature selection in svms. In *Advances in Neural Information Processing Systems*, 2002.

[55] M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[56] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.0 beta. `http://cvxr.com/cvx`, September 2013.

[57] L. Gunter and J. Zhu. Computing the solution path for the regularized support vector regression. In *Advances in Neural Information Processing Systems*, 2005.

[58] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature Extraction: Foundations and Applications*. Springer-Verlag New York, Inc., 2006.

[59] H. Zou and T. Hastie. Regularization and variable seclection via the elastic net. *Journal of the Royal Statistical Society B*, 67(2):301–320, 2005.

[60] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5:1391–1415, October 2004.

[61] B. Heisele, T. Serre, M. Pontil, T. Vetter, and T. Poggio. Categorization by learning and combining object parts. In *Advances in Neural Information Processing Systems*, 2002.

[62] T. Heskes. Solving a huge number of similar tasks: a combination of multi-task learning and hierarchical bayesian modeling. In *Proceedings of the International Conference on Machine Learning*, 1998.

[63] T. Heskes. Empirical bayes for learning to learn. In *Proceedings of the International Conference on Machine Learning*, 2000.

[64] S. J. Hwang, K. Grauman, and F. Sha. Semantic kernel forests from multiple taxonomies. In *Advances in Neural Information Processing Systems*, 2012.

[65] S. J. Hwang, F. Sha, and K. Grauman. Sharing features between objects and their attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1761–1768, June 2011.

[66] H. Daumé III. Bayesian multitask learning with latent hierarchies. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2009.

[67] Hal Daume III. Frustratingly easy domain adaptation. In *Proceedings of the Association of Computational Linguistics*, 2007.

[68] L. Jacob, F. Bach, and J. P. Vert. Clustered multi-task learning: A convex formulation. In *Advances in Neural Information Processing Systems*, 2008.

[69] A. Jain, S. V. N. Vishwanathan, and M. Varma. Spg-gmkl: Generalized multiple kernel learning with a million kernels. In *KDD*, 2012.

[70] A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan. A dirty model for multi-task learning. In *Advances in Neural Information Processing Systems*, 2010.

[71] P. Jawanpuria and J. S. Nath. Multi-task Multiple Kernel Learning. In *Proceedings of the SIAM International Conference on Data Mining*, 2011.

[72] T. Jebara. Multitask sparsity via maximum entropy discrimination. *Journal of Machine Learning Research*, 12:75–110, February 2011.

[73] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, 12:2297–2334, 2011.

[74] S. Ji, L. Sun, R. Jin, and J. Ye. Multi-label multiple kernel learning. In *Advances in Neural Information Processing Systems*, 2008.

[75] S. Ji and J. Ye. An accelerated gradient method for trace norm minimization. In *Proceedings of the International Conference on Machine Learning*, 2009.

[76] H. Kadri, A. Rabaoui, P. Preux, E. Duflos, and A. Rakotomamonjy. Functional regularized least squares classification with operator-valued kernels. In *Proceedings of the International Conference on Machine Learning*, 2011.

[77] Z. Kang, K. Grauman, and F. Sha. Learning with whom to share in multi-task feature learning. In *Proceedings of the International Conference on Machine Learning*, 2011.

[78] S. Kim and E. P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proceedings of the International Conference on Machine Learning*, 2010.

[79] J. Kivinen, M. K. Warmuth, and B. Hassibi. The p-norm generaliziation of the LMS algorithm for adaptive filtering. *IEEE Trans. Signal Processing*, 54(5):1782–1793, May 2006.

[80] M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K. R. Muller, and A. Zien. Efficient and accurate l_p-norm Multiple Kernel Learning. In *Advances in Neural Information Processing Systems*, 2009.

[81] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. l_p-norm multiple kernel learning. *Journal of Machine Learning Research*, 12:953–997, 2011.

[82] A. Kumar and H. Daume. Learning task grouping and overlap in multi-task learning. In *Proceedings of the International Conference on Machine Learning*, 2012.

[83] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[84] M. Lapin, B. Schiele, and M. Hein. Scalable multitask representation learning for scene classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[85] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, 2007.

[86] T. Levinboim and F. Sha. Learning the kernel matrix with low-rank multiplicative shaping. In *AAAI Conference on Artificial Intelligence*, 2012.

[87] B. Li, Q. Yang, and X. Xue. Transfer learning for collaborative filtering via a rating-matrix generative model. In *Proceedings of the International Conference on Machine Learning*, 2009.

[88] F. Li and C. Sminchisescu. The feature selection path in kernel methods. In *AISTATS*, 2010.

[89] F. Li, Y. Yang, and E. Xing. From lasso regression to feature vector machine. In *Advances in Neural Information Processing Systems*, 2006.

[90] J. Liu and J. Ye. Efficient l1/lq norm regularization. Technical Report arXiv:1009.4766, 2010.

[91] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

[92] K. Lounici, M. Pontil, A. B. Tsybakov, and S. van de Geer. Taking advantage of sparsity in multi-task learning. In *Proceedings of the Annual Conference on Learning Theory*, 2009.

[93] A. C. Lozano and G. Swirszcz. Multi-level lasso for sparse multi-task regression. In *Proceedings of the International Conference on Machine Learning*, 2012.

[94] C. A. Micchelli and M. Pontil. Kernels for multitask learning. In *Advances in Neural Information Processing Systems*, 2005.

[95] C. A. Micchelli and M. Pontil. Learning the kernel function via regularization. *Journal of Machine Learning Research*, 6:1099–1125, 2005.

[96] C. A. Micchelli, Y. Xu, and H. Zhang. Universal kernels. *Journal of Machine Learning Research*, 7:2651–2667, December 2006.

[97] R. S. Michalski. A Theory and Methodology of Inductive Learning. *Artificial Intelligence*, 20:111–161, 1983.

[98] S. Negahban and M. Wainwright. Phase transitions for high-dimensional joint support recovery. In *Advances in Neural Information Processing Systems*, 2009.

[99] Arkadi Nemirovski. Lectures on modern convex optimization (chp.5.4-5.5). Available at `http://www2.isye.gatech.edu/~nemirovs/Lect_ModConvOpt.pdf`, 2005.

[100] M-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1447–1454, New York, New York, June 2006.

[101] G. Obozinski, B. Taskar, and M. I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20(2):231–252, April 2010.

[102] G. Obozinski, Martin J. Wainwright, and M.I. Jordan. Support union recovery in high-dimensional multivariate regression. *Annals of Statistics*, 39:1–17, 2011.

[103] F. Orabona and L. Jie. Ultra-fast optimization algorithm for sparse multi kernel learning. In *Proceedings of the International Conference on Machine Learning*, 2011.

[104] F. Orabona, L. Jie, and B. Caputo. Online-batch strongly convex multi kernel learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 787–794, San Francisco, California, June 2010.

[105] J. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods – Support Vector Learning*, pages 185–208, 1999.

[106] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[107] L. Rivest R. Learning decision lists. *Machine Learning*, 2:229–246, 1987.

[108] A. Rakotomamonjy, F. Bach, Y. Grandvalet, and S. Canu. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

[109] A. Rakotomamonjy, R. Flamary, G. Gasso, and S. Canu. $\ell_p - \ell_q$ penalty for sparse linear and sparse multiple kernel multitask learning. *IEEE Transactions on Neural Networks*, 22(8):1307–1320, August 2011.

[110] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[111] B. Romera-paredes, H. Aung, N. Bianchi-berthouze, and M. Pontil. Multilinear multitask learning. In *Proceedings of the International Conference on Machine Learning*, 2013.

[112] S. Rosset. Following curved regularized optimization solution paths. In *Advances in Neural Information Processing Systems*, 2004.

[113] B. Scholkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

[114] B. Scholkopf, K. Tsuda, and J.-P. Vert. *Kernel methods in computational biology*. MIT Press, 2004.

[115] D. Sheldon. Graphical Multi-task Learning. Technical report, Cornell University, 2008.

[116] M. Sion. On General Minimax Theorem. *Pacific Journal of Mathematics*, 1958.

[117] L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt. Feature selection via dependence maximization. *Journal of Machine Learning Research*, 13:1393–1434, May 2012.

[118] S. Sonnenburg, G. Raetsch, C. Schaefer, and B. Schoelkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, July 2006.

[119] M. Szafranski, Y. Grandvalet, and P. M. Mahoudeaux. Hierarchical Penalization. In *Advances in Neural Information Processing Systems*, 2007.

[120] M. Szafranski, Y. Grandvalet, and A. Rakotomamonjy. Composite kernel learning. *Machine Learning*, 79:73–103, May 2010.

[121] G. Toderici, S. M. O'Malley, G. Passalis, T. Theoharis, and I. A. Kakadiaris. Ethnicity- and gender-based subject retrieval using 3-d face-recognition techniques. *International Journal of Computer Vision*, 89(2-3):382–391, September 2010.

[122] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing Features: Efficient Boosting Procedures for Multiclass Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 762–769, 2004.

[123] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, December 2005.

[124] B. A. Turlach, W. N. Venables, and S. J. Wright. Simultaneous variable selection. *Technometrics*, 47:349–363, 2005.

[125] V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[126] M. Varma and B. R. Babu. More generality in efficient multiple kernel learning. In *Proceedings of the International Conference on Machine Learning*, 2009.

[127] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *Proceedings of the International Conference on Computer Vision*, 2009.

[128] S. V. N. Vishwanathan, Z. Sun, N. Theera-Ampornpunt, and M. Varma. Multiple kernel learning and the smo algorithm. In *Advances in Neural Information Processing Systems*, 2010.

[129] G. Wang, D.-Y. Yeung, and F. H. Lochovsky. Two-dimensional solution path for support vector regression. In *Proceedings of the International Conference on Machine Learning*, 2006.

[130] G. Wang, D.-Y. Yeung, and F. H. Lochovsky. A kernel path algorithm for support vector machines. In *Proceedings of the International Conference on Machine Learning*, 2007.

[131] H. Wang, F. Nie, H. Huang, J. Yan, S. Kim, S. Risacher, A. Saykin, and L. Shen. High-order multi-task feature learning to identify longitudinal phenotypic markers for alzheimer's disease progression prediction. In *Advances in Neural Information Processing Systems*, 2012.

[132] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the International Conference on Machine Learning*, 2009.

[133] K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the International Conference on Machine Learning*, 2004.

[134] S. M. Weiss and N. Indurkhya. Lightweight rule induction. In *Proceedings of the International Conference on Machine Learning*, 2000.

[135] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for svms. In *Advances in Neural Information Processing Systems*, 2000.

[136] C. Widmer, N. Toussaint, Y. Altun, and G. Ratsch. Inferring latent task structure for multi-task learning by multiple kernel learning. *BMC Bioinformatics*, 11:S5, 2010.

[137] M. Wu and J. Farquhar. A subspace kernel for nonlinear feature extraction. In *IJCAI*, 2007.

[138] M. Wu, B. Scholkopf, and G. Bakir. Building sparse large margin classifier. In *Proceedings of the International Conference on Machine Learning*, 2005.

[139] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8:35–63, 2007.

[140] J. Ye, , S. Ji, and J. Chen. Multi-class discriminant kernel learning via convex programming. *Journal of Machine Learning Research*, 9:719–758, 2008.

[141] S. Yu, V. Tresp, and K. Yu. Robust multi-task learning with t-processes. In *Proceedings of the International Conference on Machine Learning*, 2007.

[142] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

[143] J. Zhang, Z. Ghahramani, and Y. Yang. Learning multiple related tasks using latent independent component analysis. In *Advances in Neural Information Processing Systems*, 2005.

[144] J. Zhang, Z. Ghahramani, and Y. Yang. Flexible latent variable models for multi-task learning. *Machine Learning*, 73(3):221–242, 2008.

[145] Y. Zhang and J. Schneider. Learning multiple tasks with a sparse matrix-normal penalty. In *Advances in Neural Information Processing Systems*, 2010.

[146] Y. Zhang and D. Y. Yeung. Semi-supervised multi-task regression. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, 2009.

[147] Y. Zhang and D. Y. Yeung. A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2010.

[148] P. Zhao, G. Rocha, and B. Yu. Grouped and hierarchical model selection through composite absolute penalties. *Annals of Statistics*, 37:3468–3497, 2009.

[149] P. Zhao and B. Yu. Stagewise lasso. *Journal of Machine Learning Research*, 8:2701–2726, December 2007.

[150] Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anand, and H. Liu. Advancing feature selection research. Technical report, Arizona State University, 2010.

[151] L. W. Zhong and J. T. Kwok. Convex multitask learning with flexible task clusters. In *Proceedings of the International Conference on Machine Learning*, 2012.

[152] J. Zhou, L. Yuan, J. Liu, and J. Ye. A multi-task learning formulation for predicting disease progression. In *Proceedings of the ACM Special Interest Group on Knowledge Discovery and Data Mining*, 2011.

[153] Y. Zhou, R. Jin, and S. C. H. Hoi. Exclusive lasso for multi-task feature selection. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2010.

[154] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm Support Vector Machines. In *Advances in Neural Information Processing Systems*, 2003.

Appendix B

Publications and Manuscripts

1. On p-norm Path Following in Multiple Kernel Learning for Non-linear Feature Selection.
   Pratik Jawanpuria, Manik Varma and J. Saketha Nath. In Proceedings of the International Conference on Machine Learning, 2014.
   `http://icml.cc/2014/index/article/15.htm`

2. A Convex Feature Learning Formulation for Latent Task Structure Discovery.
   Pratik Jawanpuria and J. Saketha Nath. In Proceedings of the International Conference on Machine Learning, 2012.
   `http://icml.cc/2012/papers/90.pdf`

3. Efficient Rule Ensemble Learning using Hierarchical Kernels.
   Pratik Jawanpuria, J. Saketha Nath and Ganesh Ramakrishnan. In Proceedings of the International Conference on Machine Learning, 2011.
   `http://www.icml-2011.org/papers/143_icmlpaper.pdf`

4. Multi-task Multiple Kernel Learning.
   Pratik Jawanpuria and J. Saketha Nath. In Proceedings of the SIAM International Conference on Data Mining, 2011.
   `http://epubs.siam.org/doi/pdf/10.1137/1.9781611972818.71`

5. Generalized Hierarchical Kernel Learning.
   Pratik Jawanpuria, J. Saketha Nath and Ganesh Ramakrishnan. Submitted in Journal of Machine Learning Research.

Acknowledgement

I consider myself extremely fortunate to have Prof. J. Saketha Nath as my advisor. This Ph.D. journey would not have been possible without his continuous support. In the famous episode of Yaksha-Yudhishthira[1] *samvada*, Yaksha asks Yudhishthira : "By the study of which science does man become wise?" Yudhishthira replies : "Not by studying any *sastra* does man become wise. It is by association with the great in wisdom that he gets wisdom." I will remain indebted to him for his guidance in all spheres of life.

It has been a privilege to work with Prof. Ganesh Ramakrishnan. In spite of his busy schedule, he was always available for discussions. His immense enthusiasm on a wide range of activities will always be a constant source of motivation.

I wish to express my gratitude to Prof. Sunita Sarawagi and Prof. Pushpak Bhattacharyya for their invaluable comments as my Research Progress Committee member.

I also take extreme pleasure to thank Prof. Manik Varma for his advice and support in my Ph.D. work.

My heart filled thanks to my lab-mates Ajay, Naveen and Ramkumar. We really enjoyed our discussions, which spanned on every topic under the sun other than academics. I also wish to thank Arun, Anindya, Karthik, Uma and other department friends.

I will take this opportunity to express my gratitude to all the people taking care of the administrative work, especially Mrs. Alpana Athavankar, Mrs. Homcy Varghese, Mrs. Sunanda Ghadge and Mr. Vijay Ambre in CSE, and Mrs. Gaikwad in the academic office.

I sincerely thank IIT Bombay for being a home away from home. I have relished my time here as an undergraduate and as a post-graduate. It has given me an opportunity to make a whole lot of good friends. I am proud to have spent time in the company such wonderful and

[1]Source: C. Rajagopalchari's retelling of Mahabharata

immensely talented people. Though it will be next to impossible to name them all here, I will especially like to mention the following people: Abhijit, Anandji, Anukool, Anurag, Bhushanji, Deepak, Jaydeep, Manoj, Narendra, Nilesh, Prateek, Ram Manohar, Raviraj, Saurabh, Sumit and Vishvendra.

My family have always provided their unconditional support and encouraged me to pursue my interests. Today I can see further because I am standing on their shoulders.

Pratik Jawanpuria
May 2014