Minimum Enclosing Ball-based Learner Independent Knowledge Transfer for Correlated Multi-task Learning

Fan Liu

A thesis submitted to

Auckland University Of Technology

in partial fulfillment of the requirements for the degree of

Master of Computer and Information Sciences (MCIS)

3rd February 2011

Knowledge Engineering and Discovery Research Institute (KEDRI)

Primary Supervisor: Dr. Paul S. Pang

Secondary Supervisor: Prof. Nikola Kasabov

Contents

Attestation of Authorship

"I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any other degree or diploma of a university or other institution of higher learning."

Acknowledgment

I would like to thank all the people who have helped, supported and encouraged me during my master research study.

I express my sincerest thanks to my supervisors, Dr. Paul S. Pang and Prof. Nik Kasabov who have given me great assistance, support and guidance. Dr. Pang always helps and instructs me in order that I may solve any difficulties during my research study. As a master student at Auckland University of Technology, I feel so grateful to undertake my research with my supervisor Dr. Pang at KEDRI.

I would like to express thanks to the National Institute of Communication and Technology (NICT), Japan, for providing my scholarship and an opportunity to work on their project. I would also like to acknowledge Auckland University of Technology for providing me with such a good study environment and facilities.

And last but not least, I would like to express my love and gratitude to my parents in China for their understanding, support and encouragement during my research study project at KEDRI.

Fan Liu

Auckland

3rd February 2011

Abstract

Multi-Task Learning (MTL), as opposed to Single Task Learning (STL), has become a hot topic in machine learning research. For many real world problems in application areas such as medical decision making, pattern recognition, and finance forecasting – MTL has shown significant advantage to STL because of its ability to faciliate knowledge sharing between tasks. This thesis presents our recent studies on Knowledge Transfer (KT) – the process of transferring knowledge from one task to another, which is at the core of MTL. The novelly proposed KT algorithm for correlation multi-task machine learning adapts learner independence into MTL, thus empowering any ordinary classifier for MTL.

The improvement in the learning rate of MTL as compared with STL relies on KT between tasks. Given two correlated learning tasks T^1, T^2, previous approaches to MTL are often modeled as a learner dependent model: $\mathcal{M}_{\mathcal{L}} = \mathcal{L}(T^i, KT_{\mathcal{L}}(T^i, T^j))$, for $i, j = 1, 2, i \neq j$, where $\mathcal{L}$ is a specific learner such as an k Nearest Neighbour (kNN), Support Vector Machine (SVM), or Multi-Layer Perceptron (MLP), and $\mathcal{M}_{\mathcal{L}}$ is the resultant learning model built from task T^i and transferred knowledge $KT_{\mathcal{L}}$ from T^j to T^i. Because representation of the knowledge transferred in KT_L depends on the learner, it follows that the transferred knowledge KT is incompatible for learners other than $\mathcal{L}$.

To remove the learner-dependent restriction, we propose a learner independent MTL scheme such that $\mathcal{M}_{\mathcal{L}} = \mathcal{L}(T^i, KT(T^i, T^j))$, for $i, j = 1, 2, i \neq j$, where KT is independent to the learner $\mathcal{L}$, and MTL is conducted for arbitrary learner combinations. In the proposed solution, we use Minimum Enclosing Balls (MEBs) as knowledge carriers to extract and transfer knowledge from one task to another. Since the knowledge presented in MEB can be decomposed as raw data, it can be incorporated into any learner as additional training data for a new learning task and thus improve its learning rate.

The proposed MEB-based KT is on the basis that in the feature space, the two correlated tasks share some common input data that lie on the overlapping regions

of the feature spaces in-between the two correlated tasks. The main idea is to find the correlating knowledge – overlapping regions of the two tasks – and transfer the related data regardless of the learner employed. KT is done by building a correlation space via MEBs and transferring the enclosed instances from the primary task to the secondary task. The extent of KT depends on the amount of overlapping instances between two tasks.

The effectiveness and robustness of the proposed KT is evaluated on multi-task pattern recognition (MTPR) problems derived from synthetic datasets, University of California at Irvine (UCI) datasets, and real world face datasets, using classifiers from different disciplines for MTL. We evaluate the learner independent KT model on three measures: (A) KT capability – how much knowledge can the method transfer; (B) contribution of KT – how beneficial is the transferred knowledge to the new task; and (C) adaptability of KT – how could KT adapt to learning tasks with varied relevance. It is shown that multi-task learners using KT via MEB carriers perform better than learners without-KT, and it is successfully applied to all type of classifiers.

On the other hand, in cases where the two learning tasks are semantically irrelevant, negative KT could happen to the proposed KT. For example, face recognition is not able to be supported by knowledge from hand writing recognition – simply overlapping feature domains does not work. How to discover correlated knowledge for MTL between unrelated topics remains a challenging problem and is left for future work.

Keywords — Multi-task learning, Correlated multi-task learning, Minimum Enclosing Ball, Machine Learning, Knowledge Sharing, Learner Independence, Knowledge Transfer.

Chapter 1

Introduction

This chapter introduces related work on multiple task learning, knowledge transfer, and minimum enclosing balls. MTL is described in Section 1 as a computing model for learning more than one task, enabling knowledge sharing between tasks to optimize the rate of learning individual tasks. The proposed KT for MTL is described in Section 2, where learner independence is discussed for both KT and MTL, and Section 2 also introduces the idea of MEB for KT, where MEB is used as a form of knowledge carrier, transferring knowledge from one task to another. Section 3 provides an overview of the structure of this thesis.

1.1 Multiple Task Learning

MTL is inspired by daily human activities, where multiple learning tasks might be presented at one time. In this thesis, we target multi-task classification research despite MTL is potentially applicable to clustering, regression, and many other computational researches. With a comparison to traditional single task learning, we discuss the definition of MTL, the relationship between MTL and STL, and the relationship between MTL and traditional learning algorithms.

1.1.1 Definition of MTL

Broadly speaking, MTL is everywhere in our daily life. For example, if one knows how to ride a bicycle, it will be easy for him to start to ride motorcycle, because bicycle and motorcycle riding are similar in the sense of balancing. Alternatively, if one knows how to play basketball, he should not have problem to play netball, as the rules of basketball and netball are almost the same. The goal of MTL is to model such brain-like functionality to explore the similarity between correlated activities.

MTL is a principle about learning two or more tasks one after another or at one time, while the knowledge is transferred between tasks to improve the rate of learning. The definition of MTL according to (Ozawa, Roy, & Roussinov, 2009) is "in machine learning, when more than one task learning are required, the relatedness between tasks are often modelled to facilitate task learning with the prior knowledge retained in other correlated tasks learning. This type of sequential or parallel tasks learning with KT between tasks is called multi-task learning". It is also described as correlated multi-task learning in the literature (e.g., (Abu-Mostafa, 1989; Caruana, 1997; Thrun, 1996) and (Thrun & Pratt, 1998)). Rich Caruana (Caruana, 1997) states that "multi-task learning is a model to inductive transfer using prior information from primary tasks as inductive bias to improve performance of related secondary tasks". Figure 1.1 shows an schema of correlated multiple task learning.

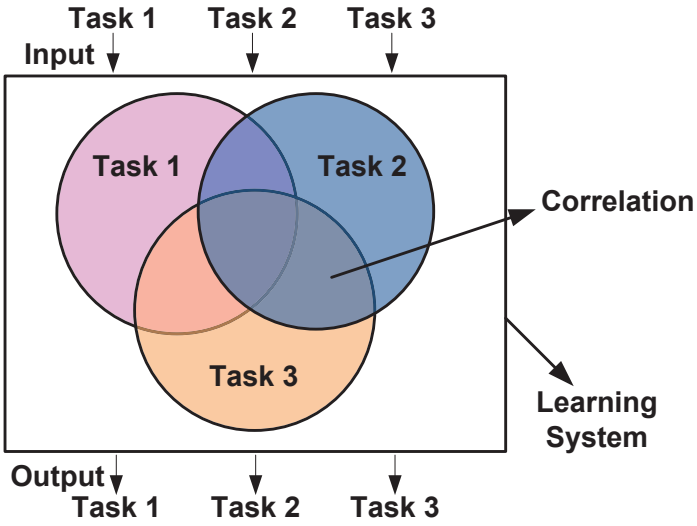


Figure 1.1: *Schematic diagram to show the idea for multiple task learning.*

1.1.2 Relationship to STL

STL refers to the building of a model for learning one task at a time, Figure 1.2 shows the concept of multiple isolated tasks for learning. In an STL, a large problem is usually decomposed into many small pieces of problem that are learned individually and separately. However, the learning system simply throws away the knowledge gained from one learning task before commencing another, rather than retaining the knowledge for future task learning. This model neglects potentially useful information which could be shared with other tasks.

In contrast to STL, MTL is not simply pooling several isolated tasks together. According to the definition above, it is different to STL in two respects. First, the tasks are not identical. For example, there could be two tasks to be performed on a face image dataset: person identification and glasses recognition. Person identification is to distinguish 271 person identity through facial image classification. Glasses recognition is to recognize whether a face is with glasses or not. Clearly, these two tasks can not appropriately be combined and treated as one single task problem. Second, the tasks are related to each other. By separating the multiple task learning problem into several independent single tasks, knowledge from one task which could be useful to the others are lost. Baxter's work (Baxter, 2000) approves the advantages of learning more than one task relies on the assumption that the tasks share a common hypothesis class. In other words, the precondition of learning more than one task relies on the relatedness between tasks.

According to the definition of Ozawa et al. (2009), "If multiple tasks are related to one another, then the learning of a particular task among them can be speeded up if knowledge is transferred from another related task that has already been learned", such transfer of knowledge in-between two correlated tasks is called knowledge transfer, and is also known as transfer learning (Pan & Yang, 2009) in the literature. Baxter (2000) has shown the principle effectiveness of KT: when there are several correlated tasks to learn, KT can mutually facilitate the learning process of the tasks.

Note that the definition of MTL also mentions KT between tasks, but this has been a "black box" procedure in previous MTL approaches (Thrun, 1996; Evgeniou & Pontil, 2004). Due to KT for MTL is developed based on a specific learning algorithm, we are not able to see what knowledge is shared and transferred between

tasks. On the other hand, STL does not allow transfer between tasks at all.

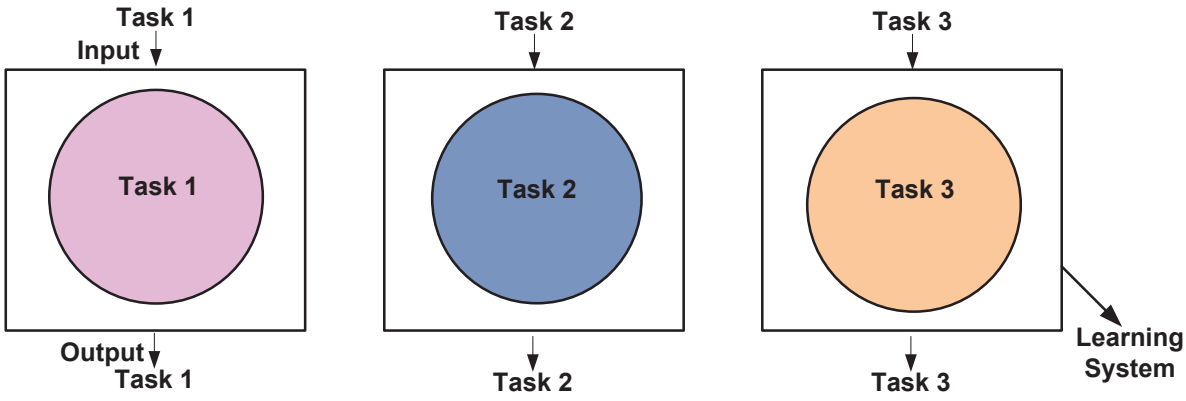


Figure 1.2: *Schematic diagram to show the idea for single task learning.*

1.1.3 Traditional MTL Algorithms

Previous MTL approaches are mostly derived from traditional machine learning algorithms such as kNN, SVM, MLP, etc. Sebastian Thrun (1996) proposes a life-long learning approach which depends on kNN to perform multiple tasks. Evgeniou et al. (2004) present a multi-task learning mechanism based on SVMs. Ando et al. (2005) provide a MTL approach derived from a Bayesian model. Ozawa et al. (2009) give a multi-task pattern recognition model based on an Artificial Neural Network (ANN). In all the above approaches, all tasks are learned by a specific learner. On the other hand, Gao et al. (Gao, Fan, Jiang, & Han, 2008) develop an ensemble framework to combine multiple classifiers by collaboration and adjustment. The ensemble framework relies on the weight assignments of component classifiers to integrate the advantages of various learning algorithms. However, the limitation with this approach is, the ensemble framework still transfers knowledge through the learners in the same way as the existing MTL approaches. The only difference is that it integrates several types of classifiers rather than using only one classifier.

1.2 Proposed Method

1.2.1 Limitations of Previous Methods

The disadvantage of most the above MTL approaches is, each method is specific to one fixed learner such as kNN or SVM, and none of them can take advantage of two or more classifiers in an unrestricted environment. This is because the KT module of the MTL has been associated with a specified learner, which makes the transferred knowledge incompatible to learners other than the original.

1.2.2 An ideal KT Approach

The core of MTL is the modelling of KT. A desirable KT-based artificial intelligent system should implement brain-like functionality. The idea behind KT is the fact that the human brain exploits knowledge from previous activities to generalize on new related activities. Take the examples of basketball and netball, and bicycle and motorcycle riding mentioned above. For previous known MTL, the task relatedness estimation and KT are conducted by building a specific learner. The procedure of KT is not transparent, as the KT approach is specialized to a fixed type of learner, and the transferred knowledge is unavailable (i.e. no usable) for a learner from another type.

In this work, we consider an interesting KT approach where the transferred knowledge depends on the correlation in-between two tasks, and is able to work with any type of learner. Figure 1.3 illustrates how a KT deals with two correlated tasks for MTL without taking the formulation of the learner into consideration. Given input data for Task 1, the task can be modelled into a set of knowledge units. For Task 2 learning, we adopt only correlated knowledge units from Task 1, and generate new knowledge for the learning of Task 2. In this way, the system addresses Task 2 with transferred knowledge from Task 1, but has no restriction on the choice of learner.

Unlike previous KT methods that are customized to one given learner, this research aims to come up with a learner independent KT approach for MTL.

The following aspects are pursued in this research:

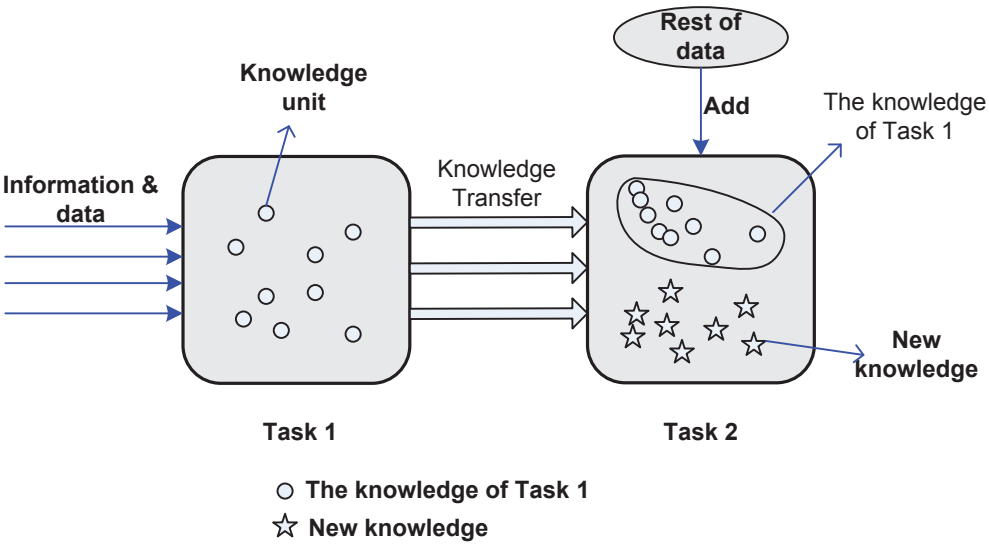


Figure 1.3: *Illustration of the concept of learner independent KT for MTL.*

1. the capability of KT, how much knowledge the method can transfer;
2. the contribution of KT, how beneficial transferred knowledge is to correlated task learning;
3. the adaptability of KT, how KT acclimatizes learning to tasks of varied relevance.

In order to design an effective KT model, two points should be taken into consideration: (1) what to transfer, that is which knowledge is shared among correlated tasks; and (2) how to transfer, that is the way to select a suitable knowledge carrier for transferring.

1.2.3 The idea of MEB-based KT

An ideal knowledge carrier has to satisfy the following criteria. (1) Flexibility in size: since the extent of KT depends on the amount of overlap/similarity between the two tasks, the carrier should have the ability to shift its scale according to the size of the correlated knowledge. (2) knowledgeable representation: the carrier itself should capture essential discriminant information. In other words, the KT from the primary

task to the secondary task should benefit the learning rate as well as the accuracy of the second learner. (3) learner independence: the knowledge carrier should not be restricted to any type of learner, the transferred knowledge can be used by any type of learner for MTL.

According to Kumar et al. MEB theory (Kumar, Mitchell, & Yildirim, 2003), given a set of points $S = \{x_1, ..., x_m\}$, where each $x_i \in \Re^d$, the minimum enclosing ball of S is the smallest ball that contains all the points in S. The MEB can be dated back to as early as 1857, when Sylvester (1857) first investigated the smallest radius disk enclosing m points on the plane. It has been applied in various areas, such as gap tolerant classifier problems, tuning SVM parameters, pre-processing (Welzl, 1991) for fast farthest neighbour query approximation and similarity search problem, testing of clustering and collision detection problem. The MEB also belongs to the larger family of shape fitting problem solvers, which attempt to find the shape (such as a slab, cylinder, cylindrical shell or spherical shell) that best fits a given set of points (Chan, 2000). Figure 1.4 illustrates data instances with multiple labels enclosed by a set of MEBs for knowledge representation. Please refer to Chapter 4 for a detailed description of MEB.

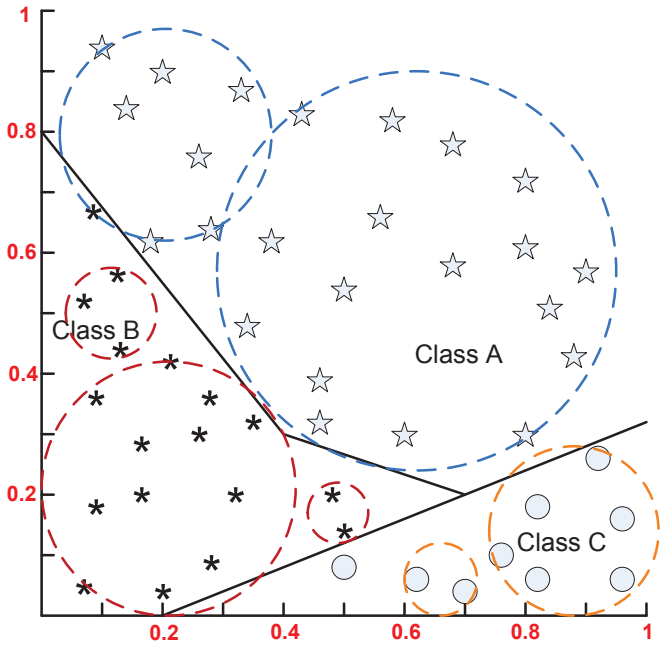


Figure 1.4: *Illustration of MEB knowledge representation for classification learning.*

In this study, we consider MEB as a potential knowledge representation method towards a learner-flexible KT approach for MTL, because of the following observations. First, MEB encloses all data points in the smallest ball no matter the size of the dataset, and the MEB itself is flexible in size. Second, MEB is a learner independent knowledge representation method. Because the enclosed data in MEB are basically uncorrelated with any learning model, knowledge contained in MEB is compatible with any learner.

The proposed method builds an assumption correlation space and transfers knowledge between tasks via MEB. It consists of two components: task relatedness interpretation and the KT algorithm. The task relatedness interpretation searches for physical task relatedness between tasks and discovers correlation areas in the feature space. The MEB-based KT algorithm extracts and transfers correlated knowledge from one learning task to another. The proposed KT method is demonstrated for MTL using different types of classifiers such as kNN, SVM, and MLP.

1.3 Structure of the Thesis

The thesis is structured as follows:

Chapter 2 contains a review of previous studies on KT methods for multi-task learning. In the review, the traditional knowledge sharing method – the learner dependent model is investigated. In the context of the learner dependent model, various KT approaches for MTL are summarized, such as inductive bias sharing, memory item sharing, and probability sharing. A learner independent model is then analyzed. The limitations of the existing KT approaches are pointed out.

Chapter 3 discusses the motivation behind the proposed research and describes relatedness interpretation as a part of the proposed KT method. The chapter begins with an introduction to the concept of learner independence in the context of KT.

Chapter 4 explains the proposed KT method using MEB-based knowledge representation. This chapter describes the principle of MEB and MEB learning from

data.

Chapter 5 demonstrates the performance of the proposed KT method. The experiment is conducted in 3 case studies: (1) the capability of KT; (2) the contribution of KT; and (3) the adaptability of KT. The MTL problems are derived from three datasets – synthetic datasets, University of California at Irvine (UCI) datasets, and real world MPEG-7 face image datasets, which are used to evaluate the performance of the three case studies, respectively.

Chapter 6 shows the results on learner independence evaluation. The MTL tasks and experimental setup are introduced, and the proposed KT method is evaluated using different types of classifiers such as kNN, SVM, and MLP.

Chapter 7 concludes the thesis and provides an overview for future works.

1.4 Denotations

Table 1.1 lists the notations that will be used in this thesis.

Table 1.1: *Notations*

Notation	Descriptions
Q	core set of MEB
C	center of MEB
C'	center of updated MEB (expansion or shrinkage)
r	radius of MEB
r'	radius of updated MEB (expansion or shrinkage)
p	point
X	the points contained in MEB
X'	X + input points of updated MEB
S	the data feature space
R	task relatedness
D	dataset (training or testing)
$\mathcal{L}$	learning system (kNN, SVM or MLP)

Chapter 2

Review of KT Methods for MTL

This chapter reviews previous approaches to KT in the context of MTL. Section 1 introduces the concept of KT in MTL. In Section 2, varieties of KT approaches are summarized into three categories: inductive bias sharing approaches, memory item sharing approaches, and probability sharing approaches. Section 3 analyzes the importance of learner independence.

2.1 Introduction

According to Pan & Yang (Pan & Yang, 2009),"KT aims to extract knowledge from one or more primary tasks and applies the knowledge to a secondary task." Thus, KT in MTL is more concerned with learning for the secondary task, than learning for both tasks. In MTL, the strength of the KT is relative to a relationship between the primary and secondary tasks (Argyriou, Maurer, & Pontil, 2008). With KT, the learning of the secondary task can be facilitated by the knowledge from the primary task; without KT, MTL may still implement learning, but correlated tasks can not be beneficial to each other as in learning with KT (Pan & Yang, 2009; Lawrence & Platt., 2004; Gao et al., 2008). This thesis focuses on KT scenarios where the learning system learns from one or more tasks and then makes predictions on a different but related task.

In previous MTL approaches, KT builds a bridge to exploit the relationship between

traditional machine learning algorithms and various transfer learning settings, in order to maintain correlated information between tasks via extracting, retaining, and transferring (Pan & Yang, 2009). Based on characteristics of various KT bridges, the different approaches to KT in the literature can be divided into the following categories (discussed in detail in Section 2.2):

- Inductive bias sharing approaches – the learning system gives a prior assumption to the previous knowledge, which is considered as inductive bias to implement KT (Mitchell, 1980).
- Memory item sharing approaches – the learning system performs KT based on the training examples stored in long-term memory, known as memory items (Ozawa et al., 2009).
- Probability sharing approaches – the learning system utilizes the hierarchical Bayesian framework to provide KT for a new task (Yu, Schwaighofer, Tresp, Ma, & Zhang, 2003).

In this thesis, we propose a novel approach to learner independent KT which is compatible with any type of learner. Currently, although a learner independent KT approach has not been widely adopted in practical MTL solutions, a similar formulation was recently introduced (Gao et al., 2008). This method relies on weight assignments from each classifier to combine the advantages of various learning algorithms such as kNN, SVM, and MLP.

2.2 Previous KT Methods

Based on the way KT is realized, we categorize KT approaches into three subcategories: inductive bias sharing approaches, memory item sharing approaches, and probability sharing approaches.

2.2.1 Inductive Bias Sharing Approaches

According to (Silver & Mercer, 2002), "The constraint of a learning system's hypothesis space, beyond the criterion of consistency with the training examples, is

called *inductive bias*". In a learning system, when a new task is learned, the learner takes a prior assumption according to previously learned knowledge and uses this prior assumption to improve learning for the secondary task. The prior assumption is known as inductive bias. "Often an inductive bias of a learning system is expressed as the system's preference for one hypothesis over another. Inductive bias is essential of the development of a hypothesis with good generalization from a practical number of examples" (Mitchell, 1980, 1997).

In the past, the inductive bias sharing method has been a major strategy in modelling KT for MTL. Silver and Mercer (2002) present the ηMTL learning framework to transfer knowledge across correlated tasks in the context of ANN. ηMTL is a knowledge-based inductive bias learning system that employs knowledge from the primary task to adjust its inductive bias. If the prior knowledge favours an inductive learner, then the knowledge is transferred from one or more primary tasks to the secondary task, which helps to reduce the hypothesis space and achieve a more accurate hypothesis from fewer training examples. Most inductive bias sharing methods learn the bias by simultaneously training models on several related tasks derived from the same data distribution and imposing constraints on their parameters. Ghosn and Bengio (2003) extend this idea for a new model that parameterizes the parameters of each task as a function of an affine manifold defined in parameter space and a point lying on the manifold. Poirier and Silver (2007) give a *context-sensitive* MTL (*cs*MTL) as a method of inductive bias transfer that uses a single output neural network and additional contextual inputs for learning multiple tasks. The *cs*MTL approach produces hypotheses that are equivalent to or better than standard inductive bias hypotheses when learning correlated tasks. Silver et al. (Silver & Mercer, 2002; Silver & Poirier, 2004; Silver, 2000) state that the inductive bias transfer method provides an indexing guide into the knowledge domain by using a deep structural measure of task relatedness, in order to improve the learning speed, using the prior internal hypothesis.

A limitation of the above approaches is that inductive bias transfer is conducted for MTL within an ANN environment. Figure 2.1 shows how an inductive bias sharing approach relies on ANN for MTL. ANN is one of the best documented methods for sharing knowledge representation among correlated tasks (Caruana, 1997; Jonathan, 1995; Intrator & Edelman, 1998). The tasks are input in parallel and share the internal knowledge representation with each other, so that all tasks can be trained

mutually and benefit each other. The hidden layer in neural network is an inductive bias, shared by all tasks for facilitating learning (Kasabov, 1996, 2007; Pang, Ozawa, & Kasabov, 2005). However, there are practical limitations to the ANNs with respect to computation time and storage space (Silver & Mercer, 2002). Theoretically, the inductive bias sharing approach is connected with a fixed type of learner, i.e., ANN, to perform MTL, so this approach is not compatible with other type of learning algorithms.

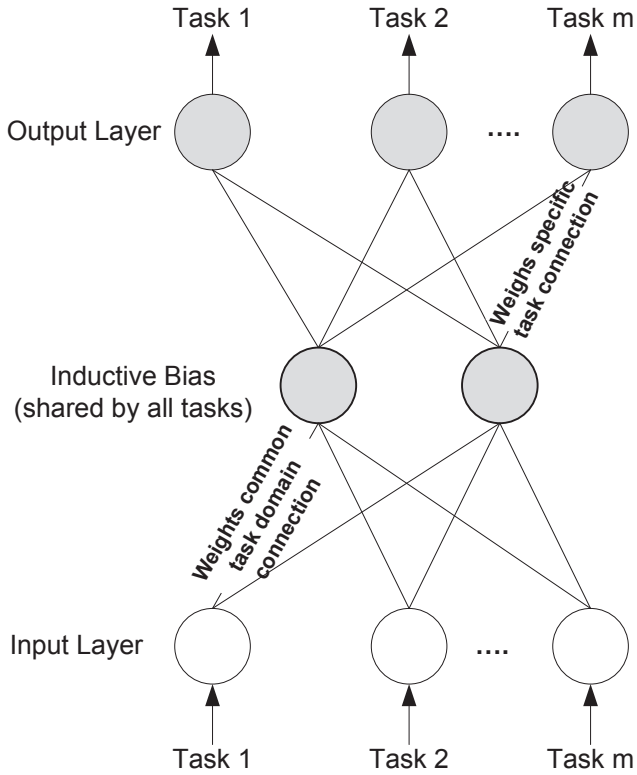


Figure 2.1: *Inductive bias sharing approach relies on prototypical 3-layer neural network for multiple task learning.*

2.2.2 Memory Item Sharing Approach

Ozawa et al. (Thrun, 1996; Ozawa et al., 2009) present a learning model for multi-task pattern recognition problems, called automated online learning, based on training samples stored in long-term memory (LTM) from previous tasks for faster learn-

ing of new correlated tasks. Here, the training samples stored in LTM are termed *memory items*.

Ozawa et al. (2009) input knowledge learned from previous tasks into memory and kept them inactive (the status flag is set to 0 for all memory items). The inactive memory items can not be used for learning a new task unless a new task is related to the existing knowledge. For a secondary task related to the primary task, the memory items corresponding to the new correlated knowledge are activated (the status flag is set to 1 for correlated memory items). The memory item sharing approach is the process of setting the status flag of active memory items to 1 to distinguish them from inactive ones and using the active memory items to implement KT.

In this memory item sharing approach, KT is achieved through activation of a set of shared memory items. This method is capable of acquiring and accumulating task knowledge, and the shared knowledge learned from tasks enhances the speed of knowledge acquisition of the secondary tasks and final classification accuracy (Ozawa et al., 2009). However, Ozawa et al. (2009) view the learner as a resource allocating network (RAN) with long term memory, which could be adapted to different tasks. In other words, the memory item transfer approach, which is based on the constraint of a RAN-LTM network structure, can only be applied to intrinsically ANN structures. Figure 2.2 shows the architecture of RAN-LTM which consists of two parts: resource allocating network and long-term memory. RAN uses a single hidden layer neural network structure and is similar to radial basis function (RBF) networks (Roy, Kim, & Mukhopadhyay, 1993; Roy, Govil, & Miranda, 1995; Roy & Mukhopadhyay, 1997; Roy, Govil, & Miranda, 1997).

Ozawa et al. (Ozawa et al., 2009) defined x, y, and z as the input vector, the output of hidden units vector, and the output of the RAN vector, respectively. The output values of the network are computed as follows:

$$y_j = exp\left(\frac{\|x - c_j\|^2}{2\sigma_j^2}\right), \quad \text{j=1,...,J} \tag{2.1}$$

$$z_k = \sum_{j=1}^{J} w_{kj} y_j + \epsilon_k, \quad \text{k=1,...,K} \tag{2.2}$$

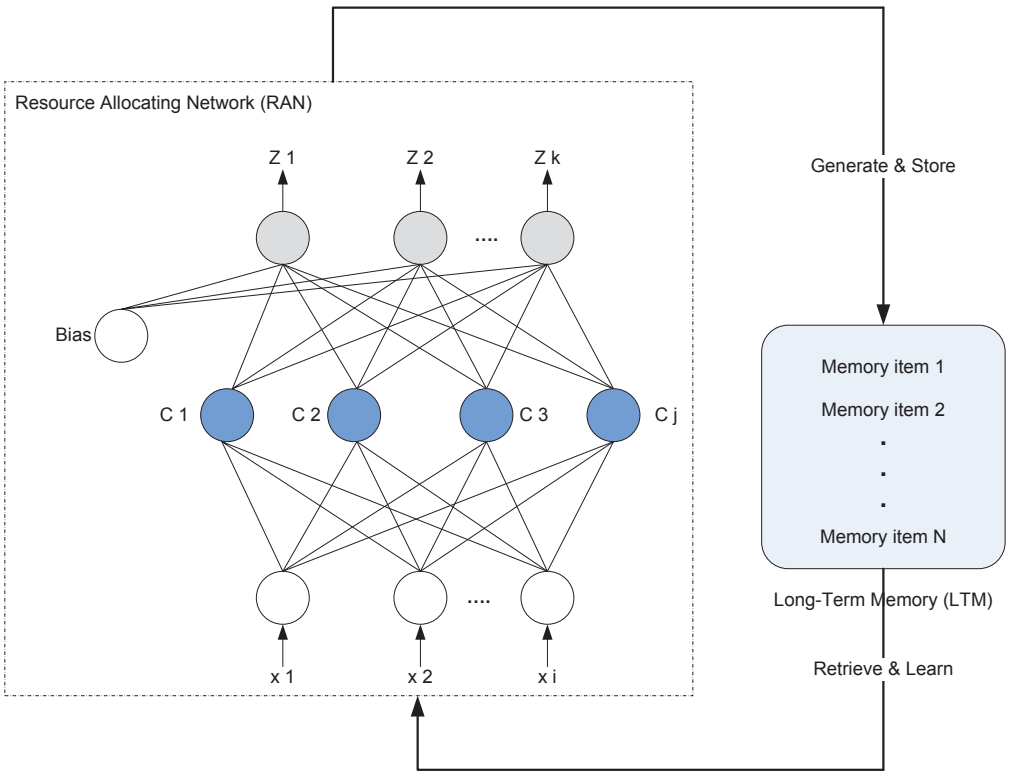


Figure 2.2: *Structure of resource allocating network with long-term memory. Memory item is the bridge of the KT to a secondary task.*

where $c_j = c_{j1}, ..., c_{jI}{}^T$ and σ_j^2 are the center and variance (width) of the jth hidden unit, w_{kj} is the connection weight from the jth hidden unit to the kth output unit, and ϵ_k is the bias of the kth output unit. The variance or width σ_j^2 is generally fixed for all RAN networks.

2.2.3 Probability Sharing Approach

Probability sharing approach is a probabilistic KT solution to conduct MTL using hierarchical Bayesian framework. According to Xue et al. (Xue, Liao, & Carin, 2007), "in hierarchical Bayesian framework, the between-task similarities are reflected in a common prior distribution placed on the model parameters of individual tasks." Typically, the common prior in a hierarchical Bayesian model is specified in a parametric form with unknown hyper-parameters, e.g., a Gaussian distribution with unknown mean and variance. Knowledge is transferred between tasks through learning those

hyper-parameters with data in all tasks. Xue et al. present a nonparametric method for the hierarchical Bayesian model. The common prior is drawn from the Dirichlet Process (DP). Ferguson (1973) proved that there is positive probability that a sample function of DP will be as close as desired to any probability function defined on the same dataset. Therefore, DP is rich enough to model the model parameters of individual tasks with arbitrarily high complexity, and flexible enough to fit well on the functional form of the prior distribution without parametric assumption.

The problem of learning from multiple related tasks is addressed in statistics and solved over hierarchal Bayesian modeling (Good, 1980). This problem has been the focus for the machine learning field over the last decade. The hierarchical Bayesian model is applied successfully to computer vision (Yu et al., 2003) and information retrieval (Evgeniou & Pontil, 2004). In Xue et al.'s experiment (2007), probability sharing method is used to transfer prior knowledge to a new task for MTL, which is based on the Dirichlet Process formulation. It includes two formulations of MTL, symmetric MTL (SMTL) formulation and asymmetric MTL (AMTL). As it avoids manipulating a mass of data from past tasks, the AMTL method provides advantageous algorithmic efficiency. The probability sharing approach is used to select sample data for learning secondary tasks. The prior probability and the posterior probability are used for the bias of KT between a primary task and the secondary task.

The probability sharing approach is different from the above two approaches in two respects. First, it is capable of dealing with new tasks without a need for knowledge storage. Second, it can possibly make use of all learned data from previous tasks for future task learning. One of its limitation is that the learning is carried out sequentially not in parallel. In other words, tasks are not available for learning at the same time. Accordingly, each model is trained separately and only a future task can benefit from previous tasks. In addition, probability sharing approach is embedded into the Bayesian model to implement multiple learning tasks. Thus, this KT method can only work under the Bayesian model for MTL, and is not applicable to other type of learning models.

Sethuraman (1994) introduces a constructive definition of the Dirichlet Process, and Ishwaran and James (2001) also characterize the DP priors with a stick-breaking representation as:

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{d_k^*}, \tag{2.3}$$

where

$$\pi_k = v_k \prod_{i=1}^{k-1} (1 - v_i). \tag{2.4}$$

For each k, v_k is drawn from a Beta distribution $Be(1, \alpha)$[1]; and another random variable d_k^* is drawn independently and simultaneously from the base distribution G_0; d_k^* and π_k represents the location and weight of the kth stick. When the $M+1$th task is given, the learning system transfers the knowledge from the previous M tasks. And thus the probability sharing approach only focuses on the new task when transferring previous task knowledge. The knowledge learned from previous tasks can be rewritten as:

$$p(d_{M+1}|d_1, ..., d_M, \alpha, G_0) = \frac{\alpha}{M+\alpha} G_0 + \frac{1}{M+\alpha} \sum_{k=1}^{k} n_k \delta_{dk^*}, \tag{2.5}$$

where α and G_0 are parameters of the variational Bayesian.

2.2.4 Learner Independence Analysis

All the aforementioned KT approaches attempt to discover the relatedness between tasks into an embedding learner/classifier for MTL. Hereinafter, this type of method is called a learner dependent KT model. In the literature, Gao et al. (2008) propose an ensemble framework, which combines several classifiers in collaboration and adjustment for MTL. In contrast to the existing KT approaches, this approach integrates several types of classifiers rather than using only one. This KT approach relies on weight assignments from each classifier to integrate the advantages of various learning algorithms such as kNN, SVM and MLP. However, since the learning system is trained independently on one fixed learner, this KT method can not break away from learners to implement KT.

According to (Gao et al., 2008), different learning algorithms have varying success and not all algorithms are equally successful in all data domains. For example, for one training set, there are usually several classification models, which can be trained and built by the different learning algorithms. Different classification models usually contain diverse knowledge, which may have different advantages over different data regions. Accordingly, classifiers in various disciplines may be effective for diverse data structures and distributions. In other words, no single classifier can perform well in all classification problems.

2.3 Summary

This chapter has reviewed different KT approaches including inductive bias approaches, memory item approaches, and probability approaches. Most previous approaches are based on the classical KT model. The limitation for such learner dependent KT approaches is that tasks are learned by one specific learner, and the transferred knowledge is not compatible with other type of learners. The importance of learner independence is also discussed - as noted by Gao et al. (2008), no single classifier can perform well in all classification problems. This inspires the certain of a learner independent KT approach as will be detailed in the next chapters.

Chapter 3

Task Relatedness and Motivation for Learner Independent KT

This chapter introduces the relatedness interpretation as an essential step of the proposed KT method and the motivation of the presented research. Section 1 presents the concept of learner independence in the context of KT. Task relatedness measurement is introduced in section 2. We seek physical task relatedness between tasks to discover correlation areas in the feature space. Section 3 defines relatedness interpretation for capturing correlated feature space between tasks, so that the volume of the correlated knowledge can be measured. Section 4 describes the motivations for learner independent KT.

3.1 Introduction

With respect to KT in MTL, most early works require that task relatedness measurement be associated to one fixed learner/classifier. The problem with the previous KT methods is that the process of transferred knowledge is not transparent. Section 2.2.4, explained that various classifiers have the advantages in addressing different data distributions, and no single classifier can perform well in all classification problems. Thus from the viewpoint of learner independence, transferred knowledge is irrelevant to the learner, which is considered to be essential for a desirable KT.

Aiming to break away from any type of learner to measure and extract correlated data transferred between tasks, in this work we developed a novel relatedness interpretation algorithm for KT, based on the assumption that the knowledge of two tasks is overlaps in the same feature space. Relatedness interpretation as a keep step of the proposed KT method is based on the assumption that all machine learning tasks are derived from the same data distribution, such that the tasks share the same problem representation and are correlated to each other. In addition, this type of KT is conducted in a physical layer to extract correlated data rather than in a functional layer, so it does not need to be embedded into a learner to implement MTL.

An additional perspective we may notice in practice is that, task relatedness conceptually refers to semantic relatedness, i.e., the relative meaning of learning topics. For example, learning to recognize facial characteristics and digital characteristics are conceptually irrelevant, but learning how to ride a bicycle and a motorcycle are relative. For measuring such semantic relatedness, feature space relatedness is often modelled for MTL as a form of distance metric, shared invariance, or similarity definition. In MTL practice, feature space relatedness is helpful only if certain semantic relatedness exits between tasks.

3.2 Measure Task Relatedness

Let T^0 be a primary task, and T^k be a secondary task with training data $D^0 = [X^0, Y^0]$, and $D^k = [X^k, Y^k]$, respectively. Theoretically, $k = 1, \ldots, m$ as there certainly exits more than one task correlated to T^0. In the presented KT research, $k = 1$ as a total of two tasks are given for KT. The relatedness R^{0k} of T^0 and T^k is typically defined over the available training samples and the hypotheses for these related tasks as,

$$R^{0k} = f_R(\mathcal{L}(D^0), \mathcal{L}(D^k), D^0, D^k), \tag{3.1}$$

where f_R can be either a static relatedness measure such as Hamming Distance or Linear Coefficient of Correlation, or a dynamic measure, between the developing hypothesis of the primary task and that of the secondary task. $\mathcal{L}$ is a learning system for MTPR, which could be any type of classifier, e.g., in ηMTL (Silver & Mercer, 2002), it is specified as an ANN. In other words, it observes functional relatedness, rather than the physical relatedness that the presented problem.

In Eq.(3.1), task relatedness is evaluated in the context of $\mathcal{L}$. The advantage of associating the task relatedness measure with a specific classifier system is, knowledge retention and transfer/use are modelled efficiently in one integrated procedure by one consistent learning system. Also, because retained knowledge is customized to $\mathcal{L}$, it is expected to be more effectively interpreted by the classifier. On the other hand, the limitation of such a learner-dependent approach is that knowledge shared between two tasks is customized into a form that suits the hypothesis model (learner), and the relatedness measure and KT are integrated into a procedure that can not be treated independently.

To empower arbitrary classifiers/learners for MTL problem solving, we propose a task relatedness measure which is independent from KT, which will in turn enable a learner-independent KT procedure. In this case, the above relatedness measure is modified as,

$$R^{0k} = f_R(D^0, D^k). \tag{3.2}$$

Excluding the influence of $\mathcal{L}$, we seek a physical task relatedness criterion which measures the "correlation" between tasks, where the "correlation" is defined as the set of samples that are mutually beneficial to perform the learning task. To this end, we have the following task correlation definition:

Definition: Given subspace S^0 spanned by a subset of D^0, cast S^0 into T^k space, if S^0 in T^k space, denoted as $S^{0\to k}$ has no 'class confliction' for T^k, then S^0 is correlated to T^k, and the correlation $C^{0\to k}$ is extracted by S^0 as,

$$\begin{array}{c} C^{0\to k}(S^0) = \arg\max_{S^0 \in S_{D^0}} |S^0| \\ \forall (\vec{x}^k, \vec{y}^k) \in S^{0\to k}, y^k \equiv c, \end{array} \tag{3.3}$$

where $|S^0|$ represents the size of S^0, and S_{D^0} is a space spanned by D^0. 'class confliction' here is interpreted as $\forall (\vec{x}^k, \vec{y}^k) \in S^{0\to k}, y^k \equiv c$, in which $(x^k, y^k) \in [X^k, Y^k]$, and c is a class label from T^k.

Applying the above definition to search for all subsets related to T^k in D^0, we have the correlation of T^0 to T^k as

$$\mathcal{C}^{0\to k} = \bigcup_{\forall S^0 \in S_{D^0}} C^{0\to k}(S^0), \tag{3.4}$$

which is reflected as a complete set correlation from T^0 to T^k. Because of the symmetry between the primary and secondary tasks, we can also have $\mathcal{C}^{k\to 0}$ from the above definition.

Fig.3.1 gives an example of correlation discovery on two synthetic classification tasks, whose data distribution and class boundaries are shown in Fig.3.1 (a) and (b), respectively. According to the above correlation definition, the theoretical task correlation can be represented as seven subregions bounded in Fig.3.1 (c), where each subregion is labelled by a pair of class labels, one for each task.

3.3 Relatedness Interpretation for MTL

Given dataset D structured as $\{X^k\}_{k=1}^M \times \{Y^k\}_{k=1}^M$ for M learning tasks $\{T^k\}_{k=1}^M$. For each learning task T^k, we have N data points $\{(x_1^k, y_1^k), (x_2^k, y_2^k), ..., (x_N^k, y_N^k)\}$ sampled from a distribution P^k on $X \times Y$. MTL is to construct hypotheses $f^1, f^2, ..., f^M$ such that $f^k(x_i^k) \approx y_i^k$ for $i \in \{1, ..., N\}$, and $k \in [1, M]$ (Ben-David & Schuller, 2003), which minimizes the following MTL cost function,

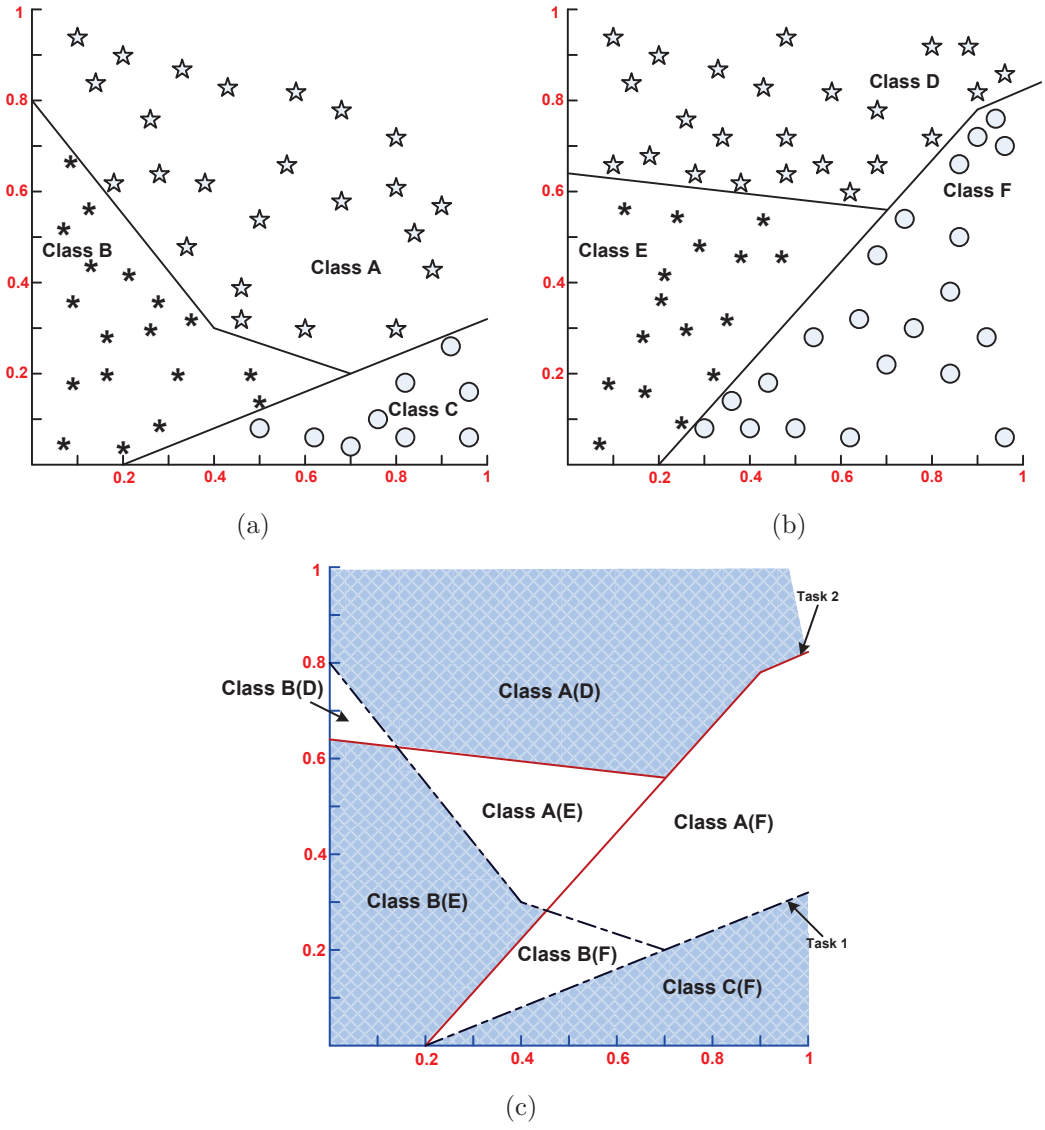
$$\mathcal{C} = \arg\min_{\mathcal{L}^k} \{\sum_{i=1}^{N} \mathcal{L}(x_i) - y_i\} \tag{3.5}$$

A straightforward solution to the above equation is to ignore task correlation, and treat the problem as M independent single tasks, and perform learning separately for each task one after another, i.e.,

$$\arg\min \{\sum_{k=1}^{M} \sum_{i=1}^{N^k} f^k(x_i^k) - y_i^k\} \tag{3.6}$$

Obviously, when learning a new and related task, it is always beneficial to take advantage of previous learning by retaining task knowledge and transferring it to the new task. Specifically, given task T^0 and T^k, KT addresses the utility of using D^0 together with D^k toward a more effective T^k hypothesis than that obtained from D^k.

According to previous studies in the literature, knowledge can be either retained in a functional form and transferred in a representational form, or retained in a



(a) (b) (c)

Figure 3.1: *An example of task correlation discovery from two synthetic classification tasks, where data points are marked as {o, *, and star}, and class labels are identified as A, B, and C for Task 1 (a), D, E, and F for Task 2 (b), respectively, and where class correlation is shown in (c).*

representational form and transferred in a functional manner (Eaton & desJardins, 2006; Eaton, 2006; Eaton, desJardins, & Stevenson, 2007). However, previous KT for MTL as discussed above mostly attaches to a specific classifier $\mathcal{L}$. Given a set of training examples, the task level learning problem concerns selecting an appropriate hypothesis,

$$f^k(x) = f(D^k, R^{0k}) = \mathcal{L}(D^k, f_R(\mathcal{L}(D^0), \mathcal{L}(D^k), D^0, D^k)),$$

where knowledge is retained and transferred through interaction with $\mathcal{L}$.

In contrast, the proposed KT employs a $\mathcal{L}$ independent knowledge retention as Eq.(3.4), so that the primary task learning can be modeled as

$$f^k(x) = f(D^k, \mathcal{C}^{0\to k}(D^0)) = f(D^k, \bigcap_{\forall S^0 \in D^0} C^{0\to k}(S^0)), \tag{3.7}$$

which excludes the affection of $\mathcal{L}$, so that any classifier can be employed for MTL.

3.4 Motivations for the Presented Research

This section introduces the motivation of the presented research. First, we note the limitations of the traditional KT approaches, then, the proposed KT method is presented to address these limitations.

3.4.1 The Limitations of Traditional KT

Given correlated tasks T^1, T^2, a sequential MTL $T^1 \to T^2$ has 3 physical operations: (1) training $\mathcal{L}_i(T^1)$; (2) extraction of knowledge by detecting task similarity/task relatedness between the two tasks, $KT_{\mathcal{L}_i}(T^1, T^2)$; (3) training $\mathcal{L}_i(T^2, KT_{\mathcal{L}_i})$ by adopting transferred knowledge.

In practice, the procedure of existing KT to MTL is not transparent, and $\mathcal{L}_i$ is a specific classifier such as kNN, SVM, or MLP. As transferred knowledge $KT_{\mathcal{L}_i}$ is $\mathcal{L}_i$ dependent, it follows that the MTL is $\mathcal{L}_i$ specific, which renders a $\mathcal{L}_j (j \neq i)$

incompatible for the MTL. For example, in the online MTL model that Ozawa et al. (2009) have recently proposed, KT is associated with a RAN-LTM structure, and thus the knowledge stored in the long term memory is not available to other types of learners. Especially, memory item sharing and probability sharing are used for a sequential MTL, in which multiple tasks are learned sequentially one after another, so that large amounts of data accumulate in the process and may cause an increment in the learning computation time (Lawrence & Platt., 2004; Thrun, 1996). Figure 3.2 shows the traditional KT model, which is associated to one fixed learner.

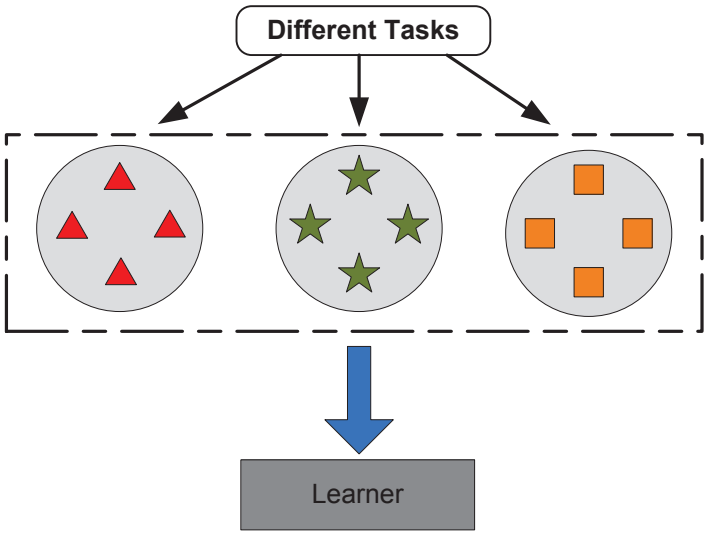


Figure 3.2: *Illustration of the traditional KT model for MTL dependent on one specific learner. The procedure of KT is not transparent.*

3.4.2 The Proposed Learner Independent KT

For STL, approaches in different disciplines may have different advantages and efforts for performing an individual task learning. The limitation of using STL for multiple tasks learning is, STL approaches forget previous task knowledge whenever commencing a new task learning. To enable MTL with knowledge share, traditional methods normally develop a KT model amongst single tasks to address knowledge forgotten problem, but this model often has to be customized to one specific classifier to suit its learning discipline. Thus in MTL literature, most MTL methods are found to involve a KT personalized to an individual classifier (Mitchell, 1980; Ozawa et al., 2009; Yu et al., 2003; Gao et al., 2008; Silver & Mercer, 2002).

Unlike previous KT for MTL, we consider an assembled MTL with KT that employs an ordinary STL approach which can be instantly enabled for optimal MTL (with-KT) by plugging in a KT module independent to any learner/classifier. This model is akin to an assembled battery that can be separated and exploited in various situations. When a battery runs out, we can replace it with another battery whatever the brand. If the use of a battery is restricted to one fixed object, then it loses functionality in regard to compatibility and independence.

Motivated by this, we propose a learner independent MTL, $\mathcal{L}_i(T^1) + KT(T^1, T^2) + \mathcal{L}_j(T^2, KT)$, where KT is independent of L_i, and MTL can be conducted by any type of learner. We use minimum enclosing balls as a form of knowledge carrier to extract and transfer knowledge from one learning task to another. The knowledge maintained in the MEB can be decomposed into raw data that could be added to any learner as additional training data for the secondary task. In other words, KT via MEBs depends on neither $\mathcal{L}_i$ nor the MEB, therefore any learner could be used for MTL by adopting $KT(T^1, T^2)$ from the primary task. Figure 3.3 presents the proposed KT model, which is independent of a specific type of learner.

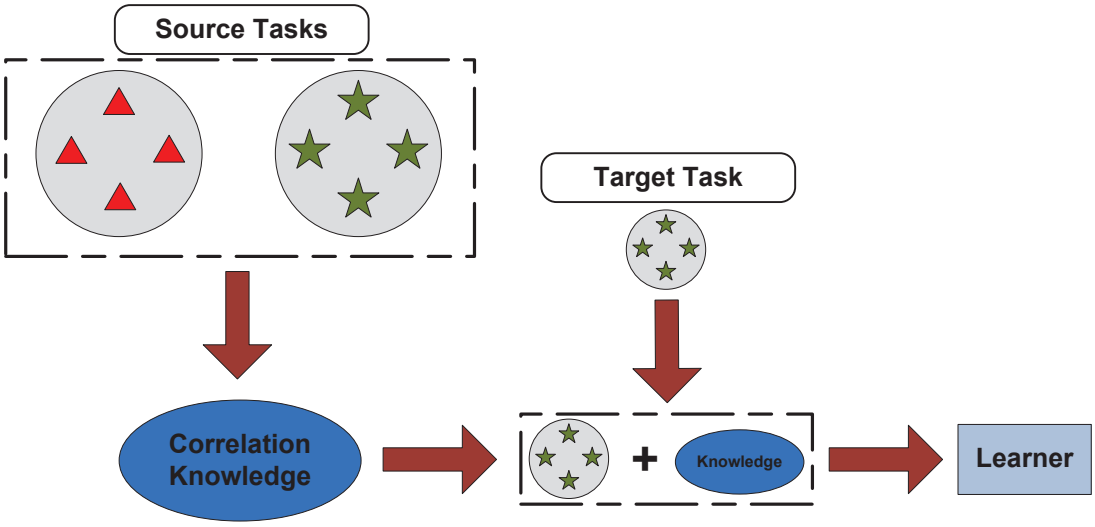


Figure 3.3: *Illustration of the proposed learner independent KT model for MTL. The procedure of KT is transparent.*

3.5 Summary

In this chapter, we have presented the motivation for a learner independent KT method, which is in contrast to traditional KT approaches which are attached to a specific classifier $\mathcal{L}$. The proposed relatedness interpretation is an essential element of the proposed KT method, which seeks a physical task relatedness measure by inter-task overlapping in the feature space. This not only discloses which data is correlated between tasks, but also excludes the influence of $\mathcal{L}$.

Chapter 4

The Proposed MEB Knowledge Transfer Algorithms

This chapter introduces the proposed knowledge transfer method, which is based on the MEB knowledge representation. Section 1 briefs the principle of MEB. Section 2 describes the learning of MEB from data. The methodology of MEB-based KT is discussed in Section 3. Finally, the proposed KT algorithm is given in Section 4.

4.1 Introduction

Megiddo (Megiddo, 1983) developed a traditional algorithm for finding exact MEBs in 1983, but it does not scale well with high dimensional datasets. Later on, Welzl (Welzl, 1991) presented an approximation algorithm for finding MEBs, where $(1+\epsilon)-approximation$ MEB can be efficiently obtained by using core sets. Additionally, Badoiu (Badoiu, 2002) found that the size of the MEB core set is independent of both the dimensionality and the size of the dataset. Based on such a size-flexible characteristic, the MEB can serve as a new knowledge representation technique towards a learner-independent KT approach.

The proposed MEB-based KT includes four major components: (1) MEB algorithm, (2) MEB expansion algorithm, (3) MEB shrinkage algorithm, and (4) correlation knowledge extraction algorithm.

4.2 MEB Representation

Given a set of points $X = \{x_1, x_2, ..., x_n\}$ with $x_i \in \Re^d$, the minimum enclosing ball of X is the smallest ball that contains all points of X (Welzl, 1991; Kumar et al., 2003; Tsang, Kwok, & Cheung, 2005). The function is denoted as $MEB(X)$ in this paper. Let $B_{c,r}$ represent the ball with center c and radius r, thus we have $B_{c,r} = MEB(X)$, and $X \subset B_{c,r}$.

To search an MEB over X with less computational cost, Welzl (Welzl, 1991) proposed the $(1+\epsilon) - approximation$ by adding a relaxation factor $1+\epsilon$ in MEB evaluation. The $(1+\epsilon)$-*approximation*, with ϵ as a small positive number, achieves MEB over X more efficiently via $MEB(Q)$. Q is a subset of X such that an expansion by a factor $(1+\epsilon)$ of its MEB, $MEB(Q) = B_{c^*,(1+\epsilon)r^*}$ contains all points of X. Q is called the core set of X, because $X \subset MEB(Q)$ as well as $Q \subset X$.

A breakthrough on achieving such an $(1+\epsilon)$-*approximation* was recently achieved by Badoiu and Clarkson (Badoiu, 2002). They used a simple iterative scheme: At the tth iteration, the current estimate $B_{c,r}$ is expanded incrementally by including the furthest point outside the $(1+\epsilon)$-ball $B_{c^*,(1+\epsilon)r^*}$. This procedure is repeated until all the points in X are covered by $B_{c^*,(1+\epsilon)r^*}$. Given an $\epsilon > 0$, the $(1+\epsilon)$-*approximation* algorithm then works as follows:

(1) Initialize x_0, c_0, and r_0.

(2) Terminate if there is no training point z such that $\varphi(z)$ falls outside the $(1+\epsilon)$-ball $B_{c_n^*,(1+\epsilon)r_n^*}$.

(3) Find z such that $\varphi(z)$ is furthest away from c_n. Set $x_{n+1} = x_n \cup \{z\}$.

(4) Find the new MEB(x_{n+1}) from (5) and set $c_{n+1} = c_{MEB(x_{n+1})}$ and $r_{n+1} = r_{MEB(x_{n+1})}$ using (3).

(5) Increment n by 1 and go back to Step (2).

Figure 4.1 gives an example of exact MEB, core set MEB, and core set MEB expansion, where the dotted circle identifies the exact MEB of the entire dataset X, and the inside solid line circle gives the exact MEB of core set Q (denoted as points

inside a square). Q does not cover the whole data points, but its $(1+\epsilon)$ expansion (the outside circle) does.

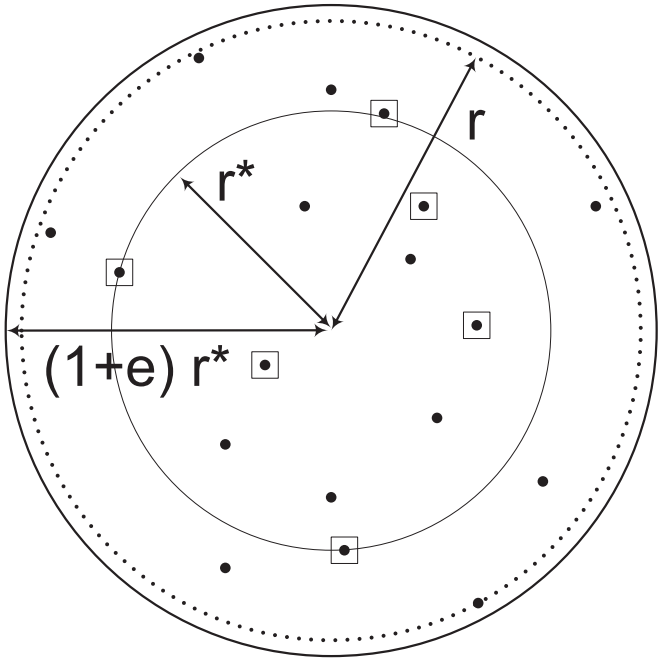


Figure 4.1: *An example of exact MEB (by radius as r), Core set MEB (by radius as r^*), and Core set MEB expansion (by radius as $(1+\epsilon)r^*$).*

4.3 Task Knowledge Transfer over MEBs

Given dataset D^0 and D^k from two correlated tasks T^0 and T^k respectively, for any subset $d^0 \subset D^0$ in one class, according to (Badoiu, 2002; Kumar et al., 2003; Tsang et al., 2005), a subspace can be spanned by modelling a minimum enclosing ball,

$$B^0_{c,(1+\epsilon)r} = MEB(d^0_i) \tag{4.1}$$

where $B^0_{c,(1+\epsilon)r}$ is able to tell whether a new input instance is enclosed by the MEB. To verify the utility of $B^0_{c,(1+\epsilon)r}$ for T^k, we cast the MEB into T^k data space, and we have

$$B^{0\to k}_{c^{0\to k},r^{0\to k}} = CAST(B^0_{c,(1+\epsilon)r}, D^0, D^k) \tag{4.2}$$

where $B^{0\to k}_{c,(1+\epsilon)r}$ is the resulting MEB casting $B^0_{c,(1+\epsilon)r}$ in T^k data space, and the $CAST$

function is implemented by calculating the casting MEB center $c^{0\to k}$ and the casting MEB radius $r^{0\to k}$, respectively.

$$c^{0\to k} = (c^0 - c^k)\frac{r^k_{max}}{r^0_{max}}, \tag{4.3}$$

and

$$r^{0\to k} = \frac{r^k_{max}}{r^0_{max}} r^0. \tag{4.4}$$

where r^0_{max} is the radius of MEB over D^0, and r^k_{max} is the radius of MEB over D^k.

The obtained $B^{0\to k}_{c,(1+\epsilon)r}$ is expected to cast a subset S^k instances in D^k. $B^0_{c,(1+\epsilon)r}$ is judged as a sharable data space by T^k, if all instances of S^k belong to one class in T^k. The instances enclosed by $B^0_{c,(1+\epsilon)r}$ are the correlation data of T^0 to T^k. In this way, given $\forall d^0 \subset D^0$, the entire sharable feature space is obtained as a merge of all MEBs that satisfy the correlation definition and the smoothness assumption (Chapelle, Scholkopf, & Zien, 2006) as: given two instances located in a high-density region, if one is enclosed in a sharable MEB, so for the other instance,

$$\begin{gathered} B^*_x = \{b^0_i\} \cup \{x\} \\ \textit{Subject to } b^0_i \in \textit{one of } D^1 \textit{ class, and } b^{0\to k}_i \in \textit{one of } D^k \textit{ class} \\ d(c, x_j) > r, d(c, x_i) < r, \textit{ and } d(x_i, x_j) < \theta. \end{gathered}$$

where θ is a distance threshold that represents the density of data distribution.

Figure 4.2 gives an illustration of the proposed MEB-based KT for a 2D synthetic dataset. As seen, an individual MEB is able to act as a knowledge carrier, casting in-between two correlated tasks with the correlation definition and being followed by the knowledge share criterion (Definition as Equation 3.3). In the case that class confliction occurs to a MEB casting to the secondary task (as shown in Figure 4.2 (b)), the MEB has the flexibility to shrink (as in Figure 4.2(c)) or expand (Figure 4.2(d)) to suit itself to the criterion of knowledge sharing.

4.4 The Proposed Algorithms

According to the theories discussed above, we summarized and developed the proposed KT algorithm, which consists of four sub-algorithms.

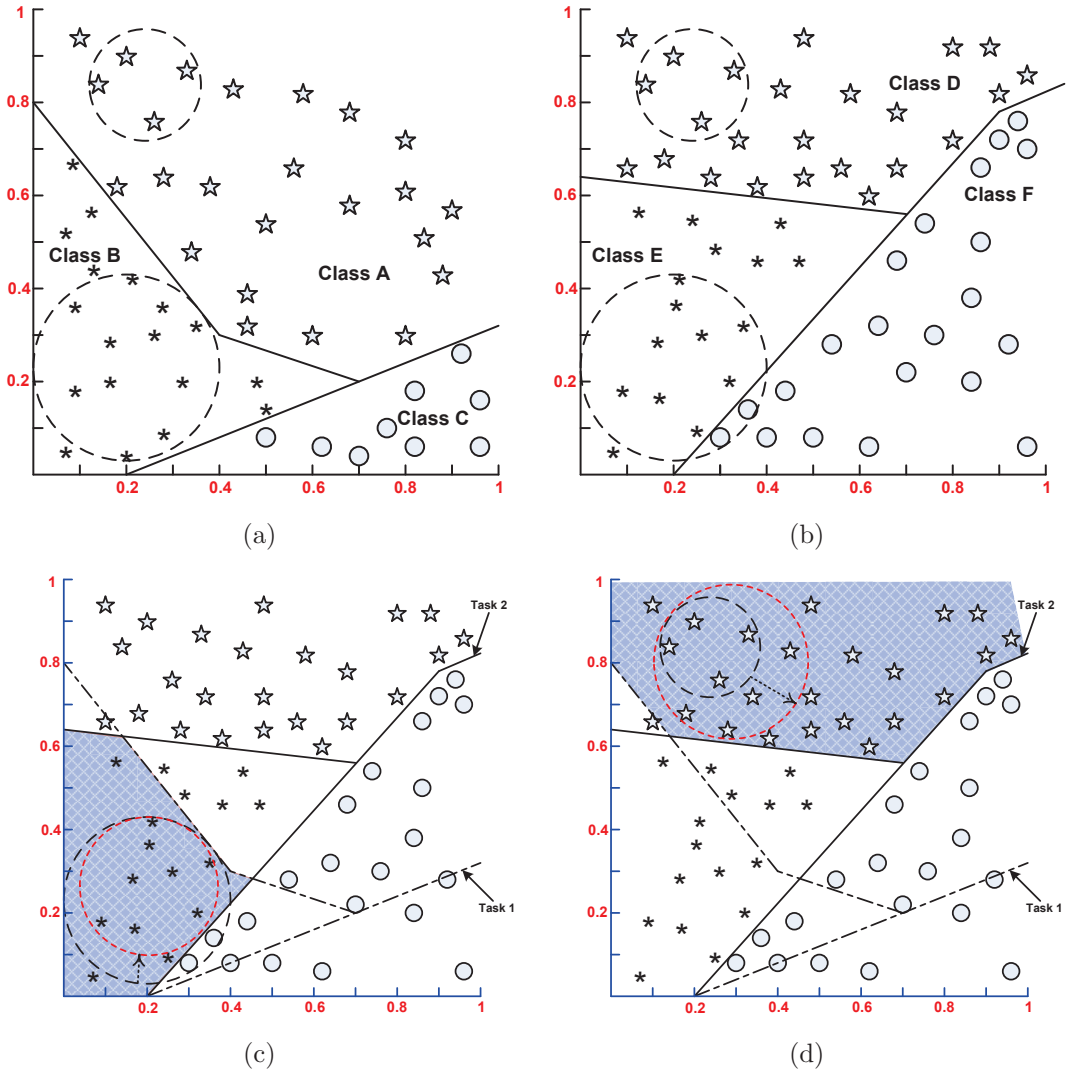


Figure 4.2: *Illustration of MEB-based KT over a 2D synthetic dataset for MTL, (a) MEBs originated in T^0, (b) results of casting MEB in T^k, (c) MEB shrink, and (d) MEB expand to suit itself to the criterion of knowledge sharing as Equation 3.3.*

(1) MEB algorithm – Algorithm 1 to compute an MEB and the core set by given a set of instances. We choose to use the two points as our initial core-set Q (Step 1, Algorithm 1). Then, the main loop (Steps 2 to 10) first computes the approximate MEB of the current subset $Q \subseteq P$. Step 4 checks if a $(1+\epsilon)$-expansion of this ball contains P. If this is the case, then the algorithm returns this expanded ball and current core-set as the solution; otherwise, the algorithm picks the furthest point in P from the center of the approximate MEB of Q, adds it to Q, and repeats the loop.

Algorithm 1 MEB Algorithm

Input: Input set of points $P \in \Re^d$, parameter $\epsilon \in (0,1)$
Output: Outputs a $(1+\epsilon)$-approximation of MEB(P) and an $O(1/\epsilon^2)$-size core-set $Q, q_i \in Q$

1: $Q \leftarrow Q_0$
2: **loop**
3: Compute $B_{c,r} = MEB(Q)$
4: **if** $P \subset B_{c,(1+\epsilon)r}$ **then**
5: Return $B_{c,r}$, Q
6: **else**
7: $p \leftarrow$ point in P maximizing distance $\|cp\|$
8: **end if**
9: $Q \leftarrow Q \cup \{p\}$
10: **end loop**

(2) MEB expansion algorithm – Algorithm 2 to expand an MEB by given a set of instances and a expansion parameter δ. We initialize the radius r' (Step 1, Algorithm 2) and enclosed instances X' (Step 2, Algorithm 2). In the main loop (Steps 3 to 7), Step 4 searches for the point x_i enclosed by a ball with the expanded radius r', adds it to X', and repeats the loop. Step 8 returns the X' to call Algorithm 1).

(3) MEB shrinkage algorithm – Algorithm 3 to shrink an MEB by given a set of instances and a shrinkage parameter δ. We initialize the radius r' (Step 1, Algorithm 3) and enclosed instances X' (Step 2, Algorithm 3). In the main loop (Steps 3 to 7), step 4 searches for the point x_i enclosed by a ball with the shrunken radius r', adds it to X', and repeats the loop. Step 8 returns the X' to call Algorithm 1).

(4) Correlation knowledge extraction algorithm – Algorithm 4 to extract correlation knowledge in-between two sets of data points. Step 1 initializes the subset V by one set of data points. The main loop (Steps 2 to 18) first computes a minimum enclosing ball by V, and then casts into another one data set space. Within the

Algorithm 2 $(1+\delta)r$ MEB Expansion Algorithm

Input: Input center c, radius r, $P \in \Re^d, x_i \in P, i = \{1, 2, ..., m\}$, parameter $\epsilon \in$ (0,1), expansion parameter δ
Output: Outputs a set of instances $X' \subseteq P$
1: $r' \leftarrow (1+\delta) * r$
2: $X' \leftarrow \phi$
3: **for** $i \leftarrow 1$ to m **do**
4: **if** $\|x_i - c\| < (1+\epsilon) * r'$ **then**
5: $X' \leftarrow X' \cup \{x_i\}$
6: **end if**
7: **end for**
8: Return $[X'] \leftarrow$ Call Algorithm 1 with input X' and ϵ

Algorithm 3 MEB Shrinkage Algorithm

Input: Input center c, radius r, subset $X \subset P \in \Re^d, x_i \in X, i = \{1, 2, ..., m\}$, parameter $\epsilon \in$ (0,1), shrink parameter δ
Output: Outputs a set of instances $X' \subset X$
1: $r' \leftarrow (1-\delta) * r$
2: $X' \leftarrow \phi$
3: **for** $i \leftarrow 1$ to m **do**
4: **if** $\|x_i - c\| < (1+\epsilon) * r'$ **then**
5: $X' \leftarrow X' \cup \{x_i\}$
6: **end if**
7: **end for**
8: Return $[X'] \leftarrow$ Call Algorithm 1 with input X' and ϵ

main loop, it consists of two sub-loops, the first sub-loop (Step 6 to 10) checks if the enclosed data of the MEB, casting from $P^{<2>}$ belongs to one class. If this is the case, then the algorithm calls the expansion algorithm (Algorithm 2) repeatedly until class confliction occurs. The second sub-loop (Step 11 to 15) checks if the enclosed data of the MEB, casting from $P^{<2>}$ are not in one class. If that is the case, the algorithm calls the shrinkage algorithm (Algorithm 3) repeatedly until the enclosed data belongs to one class. Step 16 stores all MEBs which enclose the correlated data. Step 17 finds all other data instances which are not yet extracted from $D^{<1>}$, and repeats the loop.

Algorithm 4 Correlation Knowledge Extraction Algorithm

Input: Input set of points $P^{<1>} \in \Re^d$, $P^{<2>} \in \Re^d$
Output: Output a set of minimum enclosing balls B

$V \leftarrow P^{<1>}$
while $V \neq \phi$ **do**
  $D^{<1>} \leftarrow Subset(V)$
  $B_{c,r,Q} \leftarrow$ Call Algorithm 1 with input $D^{<1>}$ and ϵ
  $D^{<2>} \leftarrow$ CAST $B_{c,r,Q}$ into $P^{<2>}$
  while $D^{<2>}$ is in one class **do**
    $B'_{c',r',Q'} \leftarrow$ Call MEB Expansion Algorithm with input $B_{c,r,Q}$
    $B_{c,r,Q} \leftarrow B'_{c',r',Q'}$
    $D^{<2>} \leftarrow$ CAST $B_{c,r,Q}$ into $P^{<2>}$
  end while
  while $D^{<2>}$ is not in one class **do**
    $B'_{c',r',Q'} \leftarrow$ Call MEB Shrinkage Algorithm with input $B_{c,r,Q}$
    $B_{c,r,Q} \leftarrow B'_{c',r',Q'}$
    $D^{<2>} \leftarrow$ CAST $B_{c,r,Q}$ into $P^{<2>}$
  end while
  $B \leftarrow [B \cup B_{c,r,Q}]$;
  $V \leftarrow V - D^{<1>}$;
end while

4.5 Summary

MEB is capable of enclosing all data points by the smallest ball in high dimensions. Usually, $(1 + \epsilon) - approximation$ MEB can be regarded as effective as the exact MEB. The proposed KT algorithm is developed based on the MEB algorithm, it contains four parts: (1) MEB algorithm for creation; (2) MEB expansion algorithm for expanding the correlation domain; (3) MEB shrinkage algorithm for shrinking the correlation domain; (4) Correlation knowledge extraction algorithm for extracting the knowledge in-between two correlated tasks.

Chapter 5

Experiments on KT for MTL

In this chapter, we evaluate the performance of the proposed KT method. Section 1 gives the experimental setup. We provide and discuss the KT results for the proposed method using three case studies: case study 1 is introduced in section 2 as KT capability, case study 2 is presented in section 3 as the contribution of KT, and section 4 describes the adaptability of KT. A discussion on KT performance is given in section 5.

5.1 Introduction

In our experiments, the proposed KT method is evaluated on a series of correlated multi-task pattern recognition problems, i.e., synthetic datasets, UCI datasets, and real world face image datasets. For each experiment, we use a 10-fold cross-validation, where accuracies are averaged over 10 runs and at each run, one tenth of the data is used as a testing set and the rest as the training set. For each cross validation, we define two classification tasks. We set one task as the primary task, and the other task is set as the secondary task. KT is always conducted from the primary task to the secondary task. The obtained correlated data is then used as additional training data for the secondary task. As the proposed KT is classifier independent, classifiers with different characteristics are applied for MTL. For comparison, we also report the results of MTL without KT. The proposed MEB-based KT algorithm is implemented

on the platform of Matlab 7.80 (R2009a), and the experiments are carried out on a PC with an Intel Core2 Duo 3.0 GH_Z CPU and 3G-byte memory.

To evaluate the robustness of the task learning system, the multiple task problems with noisy, overlapping classes are defined based on three case studies. The effectiveness of KT is evaluated on: (A) the capability of KT, to answer the question how much knowledge can the method transfer; (B) the contribution of KT, for the question that how beneficial is the transferred knowledge to a classifier for MTL; (C) the adaptability of KT, to show how KT adapts to learning tasks with varied relevance.

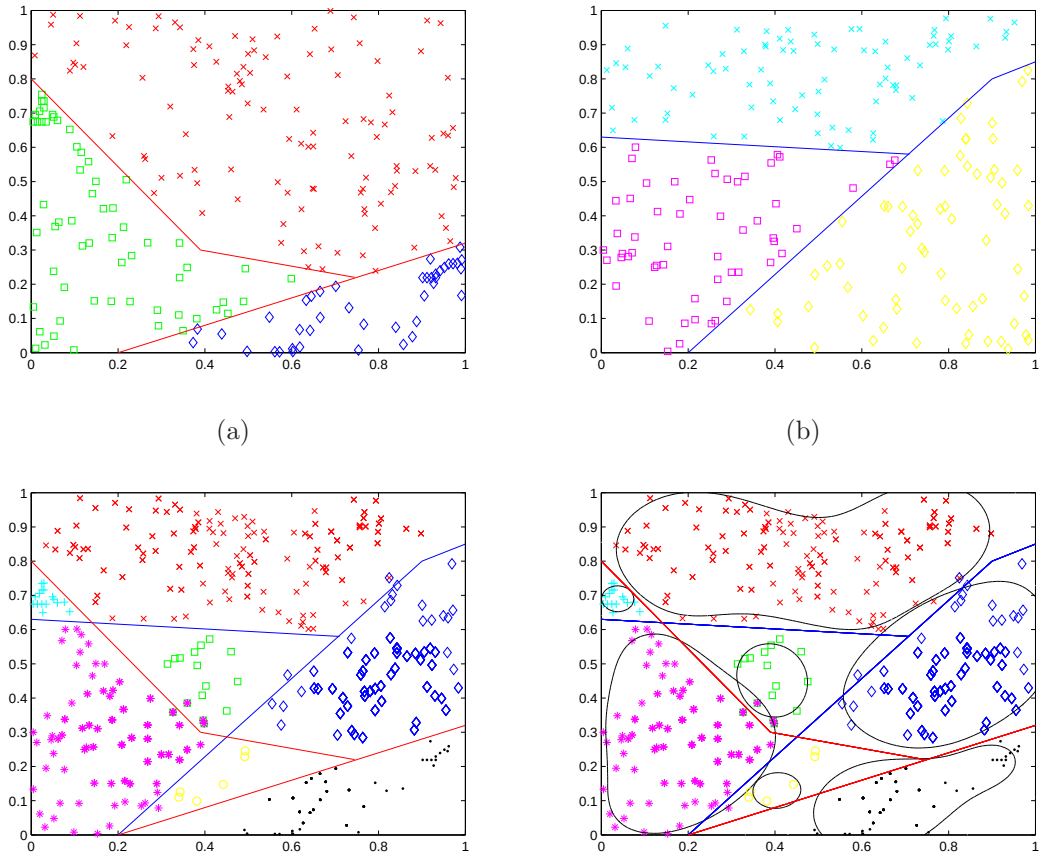
5.2 Case Study 1: KT Capability

While investigating the capability of KT, we conduct an experiment on a synthetic MTL case study. Figure 5.1 (a) and (b) depicts two synthetic classification tasks for MTL. Task 1 has 219 samples, and Task 2 has 199 samples. Task 1 and Task 2 are correlated because their data distribution overlaps.

Principally, all 418 samples should be useful for the learning of the secondary task. Figure 5.1(c) plots the correlated data (i.e. $\mathcal{C}_{12} \cup \mathcal{C}_{21}$) extracted by the proposed KT method, where the task boundaries identify seven subregions – the ground truth correlation knowledge between the two tasks. The obtained correlated data consists of 395 data points, which implies that approximately 94.5% (395/418) of the total correlated data has been extracted. Despite some correlated samples being mistakenly labelled, the usefulness of obtained correlated data on the classification of either task 1 or task 2 is confirmed by SVM subregion approximation in Figure 5.1(d).

5.3 Case Study 2: Contribution of KT to MTL

We selected five datasets from the UCI Machine Learning Repository, i.e., Segmentation, Thyroid, Vehicle, Vowel, and Yeast. The information of these five datasets is summarized in Table 5.1. Since the UCI datasets are designed for STL, we combined the original classes in different ways for MTL to create additional tasks such that every task has some relatedness to the others. The created tasks for each dataset



(a) (b)

(c) (d)

Figure 5.1: *KT for a synthetic MTL. (a) Task 1, (b) Task 2, (c) correlation data from proposed KT method, and (d) SVM approximation on the discriminability of obtained correlation data*

Table 5.1: *Information on the five UCI datasets*

Datasets	#Attributes	#Classes	#Train Samples	#Test Samples
Segmentation	19	7	210	2100
Thyroid	21	3	3772	3428
Vehicle	18	4	188	658
Vowel	10	11	528	462
Yeast	8	10	744	740

are listed in Table 5.2, where the new class labels for the newly created task are numbered after the original labels.

For each MTL topic, we do KT from Task 1 to Task 2 and Task 3, and from Task 2 back to Task 1. For performance evaluation, we conduct learning on every dataset for 100 rounds with 10-fold cross validation. We count the number of times that KT contributes positively to the secondary task learning and record the average and standard deviation of the classification accuracy difference between MTL with and without KT.

Table 5.3 shows the performance statistics for MTL on each of the five UCI datasets. As seen from the results, although MTL with KT does not always outperform that without KT, the proposed KT evaluates all 15 tasks from the five UCI datasets regardless of the type of classifier for MTL. This indicates that the proposed KT method is valid, as the contribution of KT is almost always positive for MTL.

Now we check the characteristic of classifier independence. It happens that negative KT occurs in our experiments. Because the transferred knowledge in the proposed KT is a set of raw data that reflects correlation from the primary task to secondary task, thus the transferred knowledge basically adapts to any classifiers. However, the relatedness interpretation of transferred knowledge varies among the classifiers. It seems that kernel-based classifiers, like SVM, are likely to interpret transferred knowledge better than prototype-based classifiers, like kNN – as SVM performs better than kNN on most MTL tasks.

Table 5.2: *MTPR problems created from five UCI datasets*

(a) Yeast data

Original	Task 1	Task 2	Task 3
1	1	11	13
2	2	12	14
3	3	11	15
4	4	12	16
5	5	12	16
6	6	12	16
7	7	11	16
8	8	12	16
9	9	11	16
10	10	12	16

(b) Vowel data

Original	Task 1	Task 2	Task 3
1	1	12	14
2	2	12	15
3	3	12	14
4	4	12	15
5	5	12	14
6	6	12	15
7	7	13	14
8	8	13	15
9	9	13	14
10	10	13	15
11	11	13	14

(c) Thyroid data

Original	Task 1	Task 2	Task 3
1	1	4	6
2	2	4	7
3	3	5	7

(d) Vehicle data

Original	Task 1	Task 2	Task 3
1	1	5	7
2	2	5	8
3	3	6	7
4	4	6	8

(e) Segmentation data

Original	Task 1	Task 2	Task 3
1	1	8	10
2	2	8	10
3	3	8	11
4	4	9	11
5	5	9	12
6	6	9	12
7	7	9	12

5.4 Case Study 3: Adaptability in Real World MTPR Applications

In the experiments on real world datasets, we used a MPEG-7 face database which includes 1,355 face images of 271 subjects (5 different face images per person). Each face image has a size of 56 $\times$ 46 pixels. The majority of the images are collected from AR(Purdue), AT&T, Yale, UMIST, University of Berne, and some face images are obtained from MPEG-7 news videos (Kim, Kim, & Lee, 2002; Ozawa, Pang, & Kasabov, 2006b, 2008, 2006a).

From the above face image database, we manually defined five face pattern recognition tasks as follows:

1. Person Identification (PI): to distinguish 271 subjects' identity by face image classification;

2. Pose Recognition (POR): to recognize a face in one of 5 poses: face up, face

Table 5.3: *Final probability (in percent) of classification accuracy improvement of the pattern recognition tasks based on different classifiers, i.e., kNN and SVM, for the five UCI datasets. The probability is calculated on the result of 100 runs of KT. The three values in each cell are the average probability, average accuracies increment by KT, and standard deviation in the form of (average) ± (standard deviation). These accuracies are averaged over 100 runs.*

(a) Yeast

classifier	Task 2 → Task 1	Task 1 → Task 2	Task 1 → Task 3
KNN	**100/100** (1.79% ± 1.04%)	**100/100** (1.24% ± 1.11%)	**100/100** (1.83% ± 1.15%)
SVMrbf	**75/100** (0.40% ± 1.41%)	**87/100** (0.75% ± 1.48%)	**93/100** (1.08% ± 1.55%)

(b) Vowel

classifier	Task 2 → Task 1	Task 1 → Task 2	Task 1 → Task 3
KNN	**92/100** (0.52% ± 1.01%)	**99/100** (0.42% ± 0.56%)	**100/100** (0.87% ± 0.81%)
SVMrbf	**100/100** (4.21% ± 2.88%)	**100/100** (3.25% ± 2.12%)	**100/100** (15.25% ± 6.82%)

(c) Thyroid

classifier	Task 2 → Task 1	Task 1 → Task 2	Task 1 → Task 3
KNN	**94/100** (0.14% ± 0.25%)	**99/100** (0.25% ± 0.29%)	**100/100** (0.18% ± 0.15%)
SVMrbf	**100/100** (0.22% ± 0.12%)	**100/100** (0.33% ± 0.19%)	**99/100** (0.15% ± 0.16%)

(d) Vehicle

Classifier	Task 2 → Task 1	Task 1 → Task 2	Task 1 → Task 3
KNN	**24/100** (−0.39% ± 1.66%)	**0/100** (−1.67% ± 1.31%)	**89/100** (1.00% ± 1.59%)
SVMrbf	**56/100** (0.00% ± 2.43%)	**15/100** (−0.88% ± 2.41%)	**88/100** (1.15% ± 2.24%)

(e) Segmentation

classifier	Task 2 → Task 1	Task 1 → Task 2	Task 1 → Task 3
KNN	**100/100** (0.84% ± 0.58%)	**100/100** (0.52% ± 0.54%)	**100/100** (0.92% ± 0.56%)
SVMrbf	**100/100** (14.21% ± 2.34%)	**100/100** (8.75% ± 1.67%)	**100/100** (13.18% ± 1.90%)

down, face left, face right, and face front;

3. Glasses Recognition (GLR): to recognize if a face is with glasses or not;
4. Gender Recognition (GER): to recognize the sex of subject by his/her face image.
5. Face Membership Authentication (FMA): is to authenticate the membership of an input face, when a subset of a total of 271 subjects are assigned to a member group, and the remaining people are assigned non-members of the group.

Table 5.4 (a) describes the datasets for 5 face pattern recognition tasks. The 5 tasks are correlated, because the topics are more or less linked to each other. Furthermore, they are defined on the same primary face data. However, the depth of correlation varies from task to task. For example, PI and FMA are two clearly correlated tasks, as face membership refers to a grouped personal identity. However, little correlation exists between GER and GLR, as gender is a concept irrelevant to wearing glasses.

In the experiment, we created 10 MTPR problems by combining two of the 5 tasks. Table 5.4 (b) lists the produced MTPR problems categorized by conceptual correlation depth. The top 3 problems are in a group with high correlation, the bottom 3 problems have little correlation, and the remaining 4 problems are in-between. For each MTPR problem, we conducted KT by extracting correlation data from the primary task, and appended the obtained correlation data as additional training data to the secondary learner, i.e., an RBF SVM.

For each MTPR problem, we observed the amount of correlation data transferred for MTL in Figure 5.2 (a), and evaluated the contribution to MTL in Figure 5.2 (b) in terms of the average accuracy difference between MTL with KT and without KT.

5.5 Discussion

As seen, the amount of correlation data transferred for problems with little task correlation (i.e. GLR vs POR, GER vs POR, and POR vs FMA) is approximately 10% of that from problems with high task correlation (i.e. PI vs GLR, PI vs GER, PI vs FMA), and about 40% of problems with medium task correlation (i.e. GLR

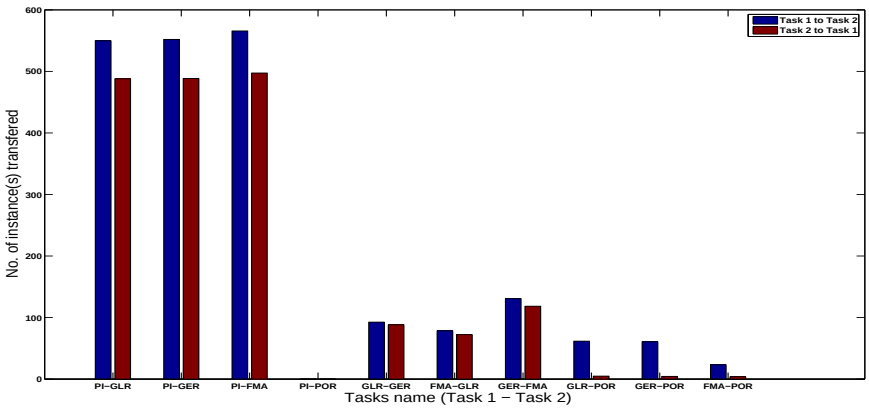
Table 5.4: *MTPR problems constructed on face image dataset*

(a) 5 single tasks defined for face pattern recognition

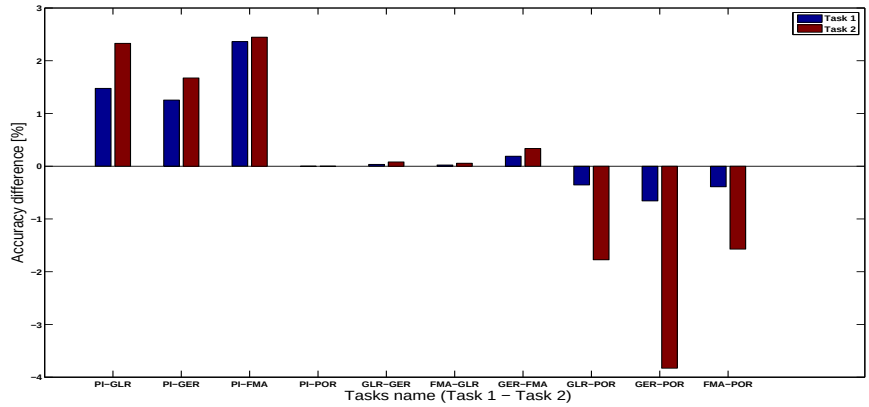
Tasks	# class	Class distribution
Person Identification (PI)	271	5 samples/class
Pose Recognition (POR)	5	Up: 16 Down: 21 Left: 36 Right: 36 Front: 51
Glasses Recognition (GLR)	2	NG: 985 WG: 370
Gender Recognition (GER)	2	M: 975 F: 380
Face Membership Authentication (FMA)	2	ME: 200 NME: 1155

(b) 10 MTPR problems created on face PR tasks in Table 5.4(a)

#Problem	Task 1	Task 2
Problem 1	PI	GER
Problem 2	PI	GLR
Problem 3	PI	FMA
Problem 4	POR	FMA
Problem 5	GLR	FMA
Problem 6	GER	FMA
Problem 7	GER	GLR
Problem 8	POR	GLR
Problem 9	POR	GER
Problem 10	PI	POR



(a)



(b)

Figure 5.2: *KT in MTPR in the case that the depth of task correlation varies. (a) Amount of correlation data transferred, and (b) average accuracy difference between MTL with KT and without KT*

vs GER, GLR vs FMA, GER vs FMA). It indicates that the proposed KT is able to adjust itself to the extent of task correlation in terms of the amount of correlation data transferred.

Also as shown in figure 5.2 (b), the contribution of KT is positive for all seven MTL problems with more or less task correlation. Negative transfer happens only to those problems with little task correlation. It follows that the proposed KT is valid as long as a conceptual correlation exists between two tasks. Although a small amount of correlation data is extracted from problems with little task correlation, their contribution to MTL may often be negative. In other words, if a small amount of correlation data is transferred but with a negative contribution, then the two tasks are conceptually irrelevant.

5.6 Summary

The effectiveness and robustness of the proposed KT is demonstrated through experiments using classifiers of different disciplines for MTL. Various multi-task pattern recognition problems derived from synthetic datasets, UCI benchmark datasets, and real world face datasets are employed in the case studies. We evaluate the learner independent KT model in three respects: (A) KT capability, to answer the question how much knowledge the method can transfer; (B) contribution of KT, for the question that how is transferred knowledge beneficial to an MTL classifier; (C) adaptability of KT, to show how the KT adapts to learning tasks with varied relevance. We found that algorithms with MEB-based KT performed better than algorithms without-KT, and MEB-based KT is successfully applied to all classifiers. However, negative KT is found in cases where two learning tasks are semantically irrelevant or have little relevance, e.g., face membership authentication and pose recognition.

Chapter 6

Learner Independence Evaluation

This chapter discusses the learner independence of the proposed KT method. Section 1 introduces the MTL tasks and experimental setup. We conduct MTL using classifiers in different discipline with the proposed KT. Sections 2–4 describe the MTL experiments with kNN , SVM, and MLP, respectively. We summarize the learner independence of the proposed KT in Section 5.

6.1 Introduction

To demonstrate the learner independence of our proposed method, we use the proposed KT algorithm for MTL on a series of correlated multi-task pattern recognition problems using different classifiers. We define MTPR problems over 5 UCI datasets in Table 5.2 and face image datasets in Table 5.4(b). We implement Algorithm 4 for KT over the tasks in Table 6.1. In the experiment, we intentionally employ classifiers with different characteristics. The classifiers examined include kNN, a prototype based classifier; SVM, a kernel based classifier; and MLP, a neural network based classifier.

For experimental setup, as the performance of the learning system could potentially be influenced by how much knowledge is transferred from one task to another, the proposed KT algorithm is evaluated by 100 runs with 10-fold cross validation for

Table 6.1: *The list of tasks for KT.*

Datasets	Defined Tasks
UCI dataset	Task 2 → Task 1
	Task 1 → Task 2
	Task 1 → Task 3
Face image dataset	PI → GLR
	GLR → PI
	PI → GER
	GER → PI
	PI → FMA
	FMA → PI

every MTL dataset. For each cross validation, we set one task as the primary task, and another task as the secondary task, and conduct KT (as Algorithm 4) always from the primary to secondary task. Then, we use the obtained correlated data as additional data for the training of the secondary task.

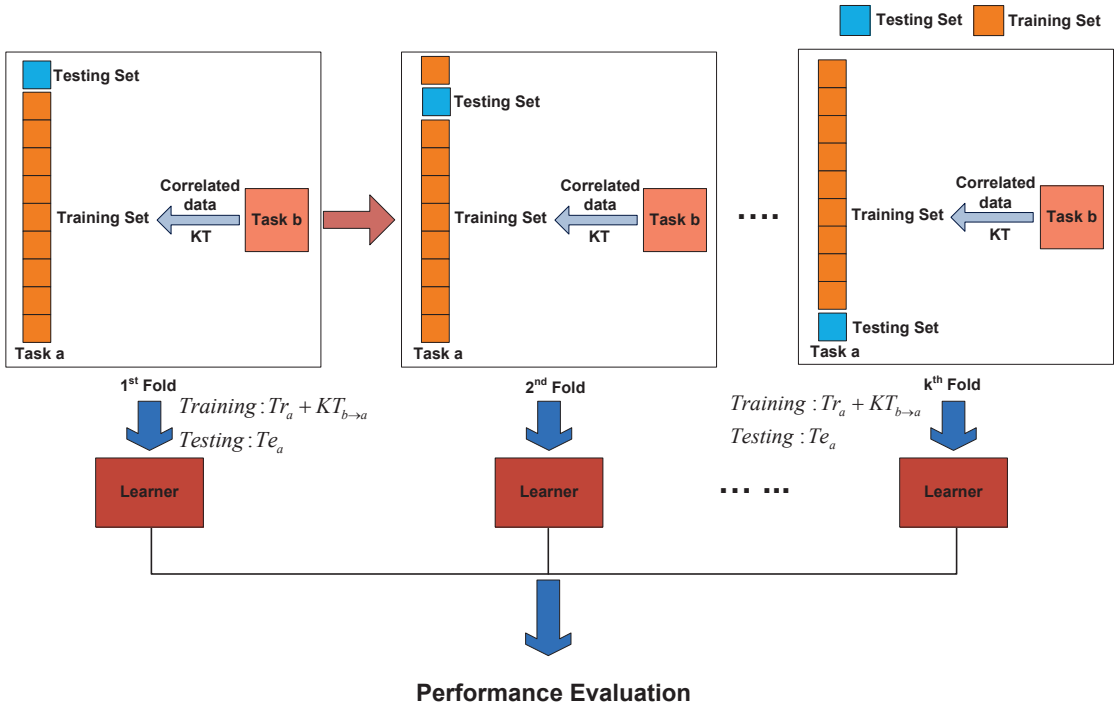


Figure 6.1: *Illustration of experiment setup.*

We test the algorithm on a set of problems to see: (1) whether the proposed KT mechanism enhances the classification accuracy, (2) how much of a contribution the proposed KT approach can make to classification accuracy, and (3) whether the proposed KT method has the steadibility to be able to be applied to any classifier. In the following, we identify these issues in detail.

KT effectiveness. It measures the rate (number of times) of successful KT which enhances classification performance compared with classification without KT. In the experiment, we conduct 100 runs with the proposed KT method for each MTL problem, and count the times that a classifier with KT from the primary task contributes positively to the secondary task. Thus it is easy to verify how many of these KT are effective.

KT contributions. We compare MTL with KT and without KT, and measure the contribution of KT by evaluating the difference in learning on classification accuracy. In the experiment, we take 100 runs for each MTPR problem, and measure the average result of classification accuracy difference between MTL with KT and without KT. The results are averaged over 100 runs.

KT steadibility. It measures the deviation value of the improved accuracy. The improved accuracy is the difference between MTL with KT and without KT. In the experiment, KT contributed accuracy is different in every round of 100, so to prove whether the proposed KT method has steadibility to a learner, standard deviation of the improved accuracy is measured. If it is high, the proposed KT does not consistently improve accuracy.

Here we introduce the results of the comparison of the three types of classifiers and show the experimental results for each task. In the following section, the concept of each classifier is introduced, then the MTL results are presented for each classifier with KT on both UCI datasets and face image datasets.

6.2 Multi-task Learning by kNN

6.2.1 k Nearest Neighbour Classifier

kNN (Thrun, 1996) is one of the most popular and simplest classifiers for data mining. kNN is a prototype based learning algorithm, which classifies objects through calculating the k nearest neighbours of incoming data, using the Euclidean distances for similarity evaluation. Since it is a simple and effective classification algorithm, it is widely used for many domains, such as pattern recognition and DNA sequencing.

Given a set of training datasets D and a query set X. For each point $x \in X$, it is classified by searching for its k nearest neighbours $(x_i, y_i) \in D$ in order to compute the output label y. Euclidean distance is the most commonly used dissimilarity measure. Figure 6.2 demonstrates a simple example of kNN classification. The set of training samples D are blue squares and green triangles, a query point, x, is shown as a red disk. x should be classified to either the first class of green triangles or the second class of blue squares. If $k = 3$, it is classified to the green triangle class, because there are two triangles and only one square in the neighbourhood (circle I). If $k = 5$, it is classified to blue square class, because there are three squares and two triangles in the neighbourhood (circle II).

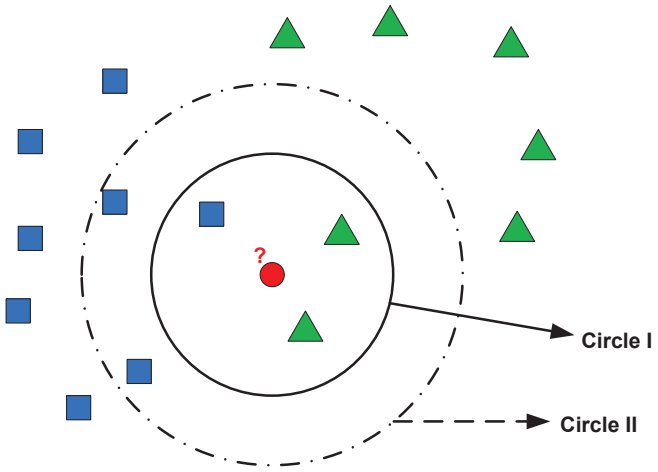


Figure 6.2: *Example of kNN classification.*

6.2.2 MTL Experiments by MEB-based KT-kNN

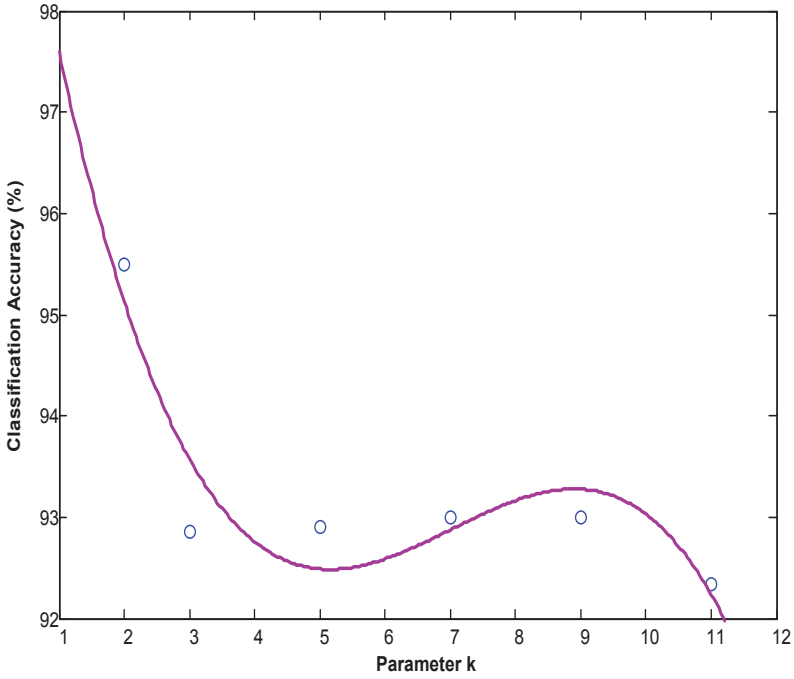
In a kNN-based MTL learning system, there are several parameters. These parameters can be divided into two groups: (1) the ones related to the kNN classifier, (2) the ones related to KT.

[Parameter for kNN Classifier]
In a kNN classifier, the parameter k has to be set, different k value might lead to different classification results. The best choice of k value depends upon data distribution. In this experiment, we use cross validation technique to select the k value. For example, as the k value must be an integer, we sequentially check the classification rate of a range of possible k values ($k \in [1, 9]$ in the experiment), in order to improve the classification accuracy by finding the proper k value.

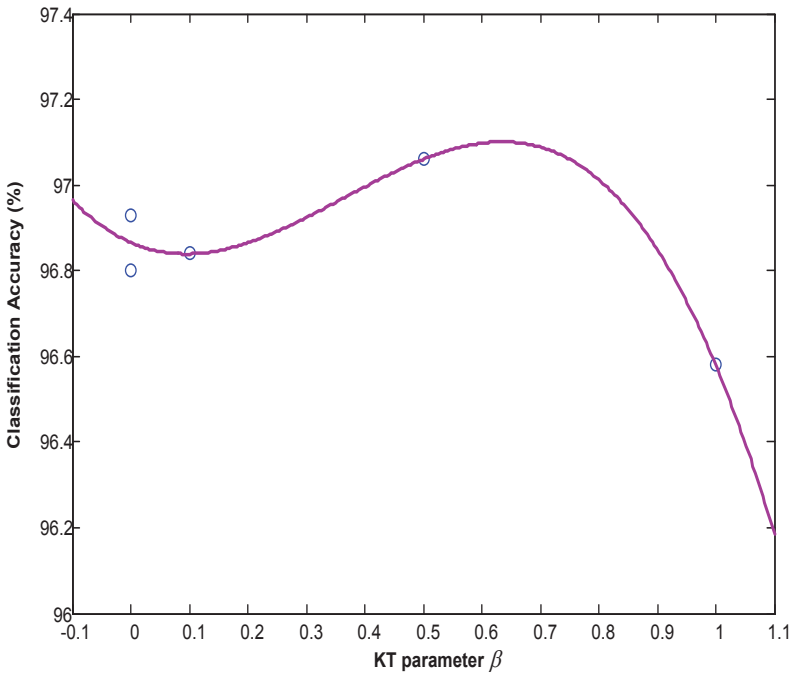
[Knowledge Transfer]
β is the control parameter to smooth the MEB, the radius of the KT region is equivalent to the radius of MEB $+\beta$.

For MEB-based KT-kNN parameter selection, k and β are determined by cross validation prediction tests on each dataset. Figure 6.3 shows the classification accuracy of KT-kNN for k and β on the Segmentation dataset. Our proposed KT method suits most of the k values, but $k = 1$ and $\beta = 0.6$ is the best choice in the experiment.

The experimental results on the kNN classifier are demonstrated in Table 6.2. Among the 5 UCI datasets, it is worth noting that most of the MTL tasks on kNN classifier achieved high performance. As can be observed in Table 6.2, the MEB-based KT-kNN persistently produces a perfect result (100% probability of accuracy improvement) on the Yeast and the Segmentation datasets. However, the MEB-based KT on the Vehicle dataset failed for Task 2 $\rightarrow$ Task 1 and Task 1 $\rightarrow$ Task 2, but the result for Task 1 $\rightarrow$ Task 3 is an acceptable 89% probability of accuracy improvement. The reason is considered to be that the data samples extracted by the MEB-based KT were few in-between Task 1 and Task 2. For the face image dataset, the proposed KT method is significant (with over 90% probability of accuracy improvement) from PI to other tasks (GLR, GER, FMA), but negtive KT occured from the other tasks to PI. The reason for this is PI holds all feature data about individuals, including the feature data included in GLR, GER, and FMA and so on. Therefore, little data is extracted by the MEB-based KT from other tasks for PI.



(a)



(b)

Figure 6.3: *Parameter selection for MEB-based KT-kNN. The figures show the classification accuracies for the Segmentation dataset by different values of parameter k and β. The best result is obtained when k= 1 and $\beta = 0.6$. (a) kNN parameter k. (b) Smoothing value of KT region β, KT is from Task 2 to Task 1.*

Table 6.2: *Final probability (in percent) of classification accuracy improvement of the pattern recognition tasks based on kNN classifier for the five UCI datasets and the face image dataset. The probability is measured after the completion of 100 KT runs. The three values in each cell are the average probability, average rising accuracy between with-KT and without-KT, and standard deviation in the form of (average) ± (standard deviation).*

(a) UCI Datasets

UCI Datasets	Task 2 → Task 1	Task 1 → Task 2	Task 1 → Task 3
Yeast	**100/100** (1.79% ± 1.04%)	**100/100** (1.24% ± 1.11%)	**100/100** (1.83% ± 1.15%)
Vowel	**92/100** (0.52% ± 1.01%)	**99/100** (0.42% ± 0.56%)	**100/100** (0.87% ± 0.81%)
Thyroid	**94/100** (0.14% ± 0.25%)	**99/100** (0.25% ± 0.29%)	**100/100** (0.18% ± 0.15%)
Vehicle	**24/100** (−0.39% ± 1.66%)	**0/100** (−1.67% ± 1.31%)	**89/100** (1.00% ± 1.59%)
Segmentation	**100/100** (0.84% ± 0.58%)	**100/100** (0.52% ± 0.54%)	**100/100** (0.92% ± 0.56%)

(b) Face Image Datasets

	PI → GLR	GLR → PI
PI vs GLR	**100/100** (1.20% ± 0.85%)	**0/100** (−0.72% ± 0.76%)
	PI → GER	GER → PI
PI vs GER	**91/100** (0.41% ± 0.63%)	**3/100** (−0.75% ± 0.89%)
	PI → FMA	FMA → PI
PI vs FMA	**91/100** (0.01% ± 0.01%)	**4/100** (−0.01% ± 0.01%)

6.3 Multi-task Learning by SVM

6.3.1 Support Vector Classifier

The original support vector machine (SVMs) by Boser et al. (Boser, Guyon, & Vapnik, 1992) was derived from the generalized portrait algorithm invented earlier by Vapnik and Lerner (Vapnik & Lerner, 1963). Since then, SVM has become one of the most popular classification methods based on statistical learning (Evgeniou & Pontil, 2004), and it is widely used in the machine learning area as a powerful approach for solving problems with high dimensional classification. SVM is an approach to maximize the margin ($|L1 - L2|$) for separating classes in high dimensional data space. Figure 6.4 shows a simple example of an SVM optimal separating hyperplane for a two-class classification.

We assume that we have a training set $D = (x_1, y_1), (x_2, y_2), ..., (x_i, y_i)$, where $x \in \Re^d$ is the d-dimentional input vector, $y \in \{-1, 1\}$ is a class label, $i \in \Re$, for which there exists a norm vector $w \in \Re^d$ with $\|w\|^2 = 1$ and a bias parameter $b \in \Re$ such that

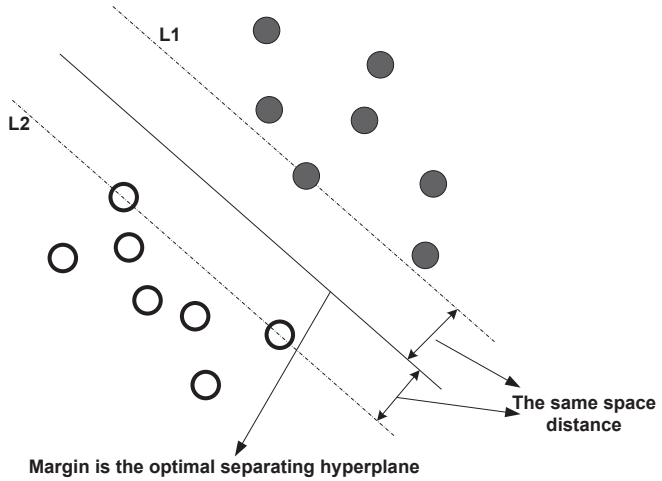


Figure 6.4: *Example of SVM optimal separating hyperplane for two-class classification case. The black disks and the white disks are samples from two different classes.*

$$\begin{aligned} &\langle w, x_i \rangle + b > 0, \quad for\ all\ i\ with\ y_i = +1, \\ &\langle w, x_i \rangle + b < 0, \quad for\ all\ i\ with\ y_i = -1. \end{aligned} \tag{6.1}$$

In other words, the affine linear hyperplane described by (w, b) perfectly separates the training set D into the two groups $(x_i, y_i) \in D : y_i = +1$ and $(x_i, y_i) \in D : y_i = -1$. According to Vapnik's SVM theory (Cortes & Vapnik, 1995), a non-linear decision function $f(x)$ of SVM is the form of

$$f(x) = sign\,(w \cdot \Phi(x) + b) \tag{6.2}$$

where "$\cdot$" means a dot product and $\Phi(x)$ refers to an implicitly mapped vector in the feature space induced by the kernel function $k(x, \acute{x}) = \langle \Phi(x), \Phi(\acute{x}) \rangle$.

We can convert the above classification problem into an optimization problem as:

$$\begin{aligned} \text{minimize} \quad & \langle w, w \rangle \quad && over\ w \in \Re^d, b \in \Re \\ \text{subject to} \quad & y_i(\langle w, x_i \rangle + b) \geq 1 \quad && i = 1, ..., n. \end{aligned} \tag{6.3}$$

To resolve the optimization problem in 6.3, we first map the input data $x_i, ...x_n$ into a feature space by a typically non-linear mapping $\Phi : X \rightarrow H_0$. Then, the generalized

portrait algorithm is applied to the mapped dataset $((\Phi(x_1), y_1), ..., (\Phi(x_n), y_n))$. To training an SVM, a quadratic optimization problem need to be solved as:

$$\begin{array}{rll} \text{minimize} & \frac{1}{2}\langle w, w\rangle + C\sum_{i=1}^{n}\xi_i & for\ w \in H_0, b \in \Re, \xi \in \Re_n \\ \text{subject to} & y_i(\langle w, \Phi(x_i)\rangle + b) \geq 1 - \xi_i, & i = 1, ..., n \\ & \xi_i \geq 0, & i = 1, ..., n, \end{array} \tag{6.4}$$

where $C > 0$ is a positive constant which is used to balance the objective function and ξ is slack variable which controls the trade-off between the objective function and the number of miss classifications. These two parameters can be typically determined by cross validation. We can rewrite the equation 6.4 as:

$$L(y_i, \langle w, \Phi(x_i)\rangle + b) = \begin{cases} 0 & \text{if } |y_i - \langle w, \Phi(x_i)\rangle + b| \leq \xi \\ |y_i - \langle w, \Phi(x_i)\rangle + b| - \xi & \text{otherwise,} \end{cases} \tag{6.5}$$

where L is the loss function. This quadratic optimization problem need to be solved in a high or infinite dimensional feature space H_0. It is easy to use Lagrange multipliers approach to obtain a dual optimization of the problem in 6.4,

$$\begin{array}{rll} \text{minimize} & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \langle \Phi(x_i), \Phi(x_j)\rangle & \alpha \in [0, C]^n \\ \text{subject to} & \sum_{i=1}^{n} y_i \alpha_i = 0, & \end{array} \tag{6.6}$$

Let $(\alpha_i^*, ..., \alpha_n^*)$ be the solution of the equation 6.6, the solution (w_D^*, b_D^*) of the equation 6.4 can be computed as

$$w_D^* = \sum_{i=1}^{n} y_i \alpha_i^* \Phi(x_i)$$

and

$$b_D^* = y_j - \sum_{i=1}^{n} y_i \alpha_i^* \left\langle \Phi(x_i), \Phi(x_j) \right\rangle ,$$

where j is an index with $0 < \alpha_j^* < c$. w_D^* only depends on the samples x_i whose weights satisfy $\alpha_i^* \neq 0$. Geometrically, this means that the affine hyperplane described by (w_D^*, b_D^*) is only supported by these $\Phi(x_i)$, and hence the corresponding data points (x_i, y_i) are called support vectors (Cortes & Vapnik, 1995). The decision function $f_{w_D^*, b_D^*}(x)$ can be written by the constructed affine hyperplane as,

$$f_{w_D^*, b_D^*}(x) = \langle w_D^*, \Phi(x) \rangle + b_D^* = \textstyle\sum_{i=1}^{n} y_i \alpha_i^* \left\langle \Phi(x_i), \Phi(x) \right\rangle + b_D^*, \quad x \in X. \tag{6.7}$$

In both the dual optimization problem 6.6 and the decision function 6.7 only inner products of Φ occur. Therefore, instead of computing the feature map, it is sufficient to know the function $\langle \Phi(\cdot), \Phi(\cdot) \rangle : X \times X \to \Re$. According to the theory of Vapnik et al. (Cortes & Vapnik, 1995) , there are some cases which we can compute the function $\langle \Phi(\cdot), \Phi(\cdot) \rangle$ without knowing the feature map Φ itself, such as Gaussian RBF kernels. The kernel function that satisfies Mercer's conditions can be written as:

$$k(x, x') = \left\langle \Phi(x), \Phi(x') \right\rangle , \quad x, x' \in X. \tag{6.8}$$

We can use kernels directly instead of explicit feature vectors in the feature space in algorithms where only the inner product of the feature map but not the feature map itself are involved. Some well known kernel functions are listed below (Table 6.3).

Table 6.3: *SVM Kernels*

SVM Kernels	
Polynomial (homogeneous):	$k(x, \acute{x}) = (x \cdot \acute{x})^d$
Polynomial (inhomogeneous):	$k(x, \acute{x}) = (x \cdot \acute{x} + 1)^d$
Radial Basis Function (RBF):	$k(x, \acute{x}) = exp(-\gamma \|x - \acute{x})\|^2)$, for $\gamma > 0$
Gaussian Radial basis function:	$k(x, \acute{x}) = exp(-\frac{\|x - \acute{x})\|^2}{2\sigma^2})$
Sigmoid:	$k(x, \acute{x}) = tanh(\kappa x \cdot \acute{x} + c)$ for some (not every) $\kappa > 0$ and $c < 0$

To evaluate the validity of MEB-SVM, we compare SVM with MEB-based KT

against a SVM without KT. The Support Vector Classification (SVC) employs transferred knowledge $D^{'}$ from primary tasks to secondary task in addition to the training data D for secondary task.

$$f(x) = sign(w \cdot \Phi([x \cup D^{'}]) + b) \tag{6.9}$$

6.3.2 MTL Experiments by MEB-based KT-SVM

In the SVM learning system, several parameters have to be set. As shown below, those parameters can be divided into two groups: (1) the ones related to the SVM classifier, (2) the ones related to KT.

[SVM Classifier]
g – gamma: the constant in the Radial Basis Function Kernel function.
c – cost: the penalty parameter from the original SVM formulation.

[Knowledge Transfer]
β – It is the control parameter for smoothing MEB, the radius of KT region = the radius of MEB $+\beta$.

Figure 6.5 shows the final classification results of SVM with different g, c and β parameters on the Segmentation dataset. $g = 0.001, c = 8$ is the best choice for SVM parameter. We use the same cross validation technique to determine the SVM classifier parameters and β parameter for each dataset. After parameter selection, the experimental results on SVM classifier are demonstrated as follows.

For MTL experiments by SVM classifier, the generalization performance for most UCI datasets is acceptable, except the Vehicle – negative KT is occurred same as by kNN classifier. The result for Yeast dataset is not as well as MEB-based KT by kNN. For the face dataset, the proposed method produces probability of accuracy improvement with 91% and 97%, from PI to GLR and from PI to FMA, respectively. The MEB-based KT is most successful. See Table 6.4 for additional results.

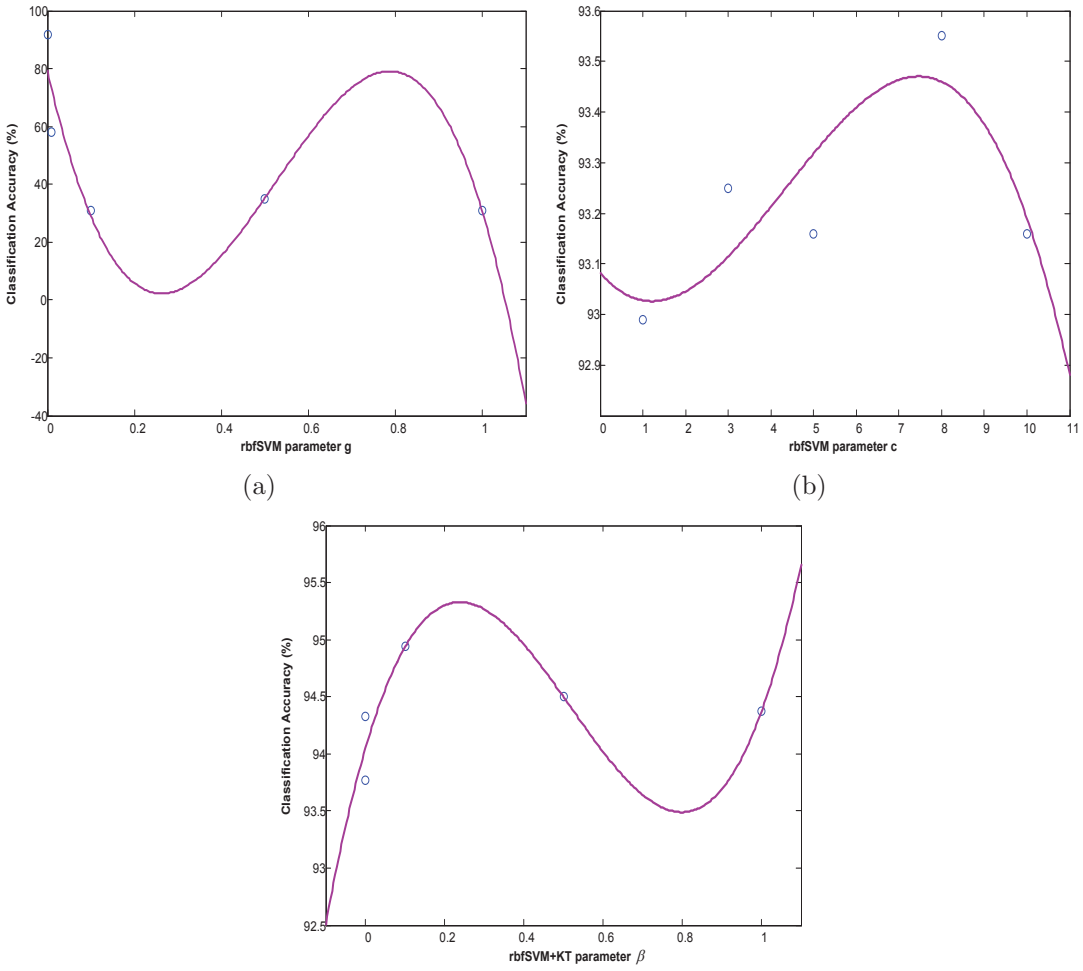


Figure 6.5: *MEB-based KT-rbfSVM for different model parameters: The classification accuracy for the Segmentation dataset with different values of parameters g, c and, β. The best result is obtained when* $g = 0.001, c = 8$, *and* $\beta = 0.2$*(a) The gamma of kernel function g. (b) The margin parameter c of C-SVC. (c) Smoothing value of KT region β, KT is from Task 2 to Task 1.*

Table 6.4: *Final probability (in percent) of classification accuracy improvement of the pattern recognition tasks based on SVM classifier for the five UCI datasets and the face image dataset. The probability is measured after the completion of 100 runs KT. The three values in each cell are the average probability, average rising accuracy between with-KT and without-KT, and standard deviation in the form of (average) ± (standard deviation).*

(a) UCI Datasets

UCI Datasets	Task 2 → Task 1	Task 1 → Task 2	Task 1 → Task 3
Yeast	**75/100** (0.40% ± 1.41%)	**87/100** (0.75% ± 1.48%)	**93/100** (1.08% ± 1.55%)
Vowel	**100/100** (4.21% ± 2.88%)	**100/100** (3.25% ± 2.12%)	**100/100** (15.25% ± 6.82%)
Thyroid	**100/100** (0.22% ± 0.12%)	**100/100** (0.33% ± 0.19%)	**99/100** (0.15% ± 0.16%)
Vehicle	**56/100** (0.00% ± 2.43%)	**15/100** (−0.88% ± 2.41%)	**88/100** (1.15% ± 2.24%)
Segmentation	**100/100** (14.21% ± 2.34%)	**100/100** (8.75% ± 1.67%)	**100/100** (13.18% ± 1.90%)

(b) Face Image Datasets

	PI → GLR	GLR → PI
PI vs GLR	**91/100** (0.50% ± 1.14%)	**53/100** (−0.02% ± 1.29%)
	PI → GER	GER → PI
PI vs GER	**72/100** (0.24% ± 0.92%)	**40/100** (−0.15% ± 1.23%)
	PI → FMA	FMA → PI
PI vs FMA	**97/100** (0.50% ± 0.70%)	**48/100** (−0.08% ± 1.21%)

6.4 Multi-task Learning by MLP

6.4.1 Multi-layer Perceptron Classifier

Multi-layer perceptron (MLP) (Jusman et al., 2009) is a feed-forward artificial neural network model, in which the tasks are inputted in parallel and share the internal knowledge representation with each other. An MLP network consists of several layers of neurons: the input layer is the first layer, and the output layer is the last layer, remaining layers are called hidden layers. There are complete connections between the nodes in successive layers but there is no connection between neurons within the same layer. Every node, except the input layer nodes, computes the weighted sum of its inputs and apply a sigmoidal function to compute its output, which is then transmitted to the nodes of the next layer (Haykin, 1999). The objective of MLP algorithm is to set the connection weights such that the disagreement between the network output and the target output is minimized.

For pattern classification, MLP adjusts the free weights during supervised training and partition the input space using linear hyperplanes. To separate various classes, MLP estimates a function in the form which partitions the input space into polyhe-

dral sets or regions so that each point in the domain is assigned to one out of the m classes of Y. Each node has an associated hyperplane to partition the input space into two half-spaces. The combination of the individual, linear node-hyperplanes in additional layers allows a stepwise separation of complex regions in the input space, generating a complex decision boundary to separate the different classes (Haykin, 1999). Figure 6.6 shows a simple mechanism of MLP-based pattern recognition.

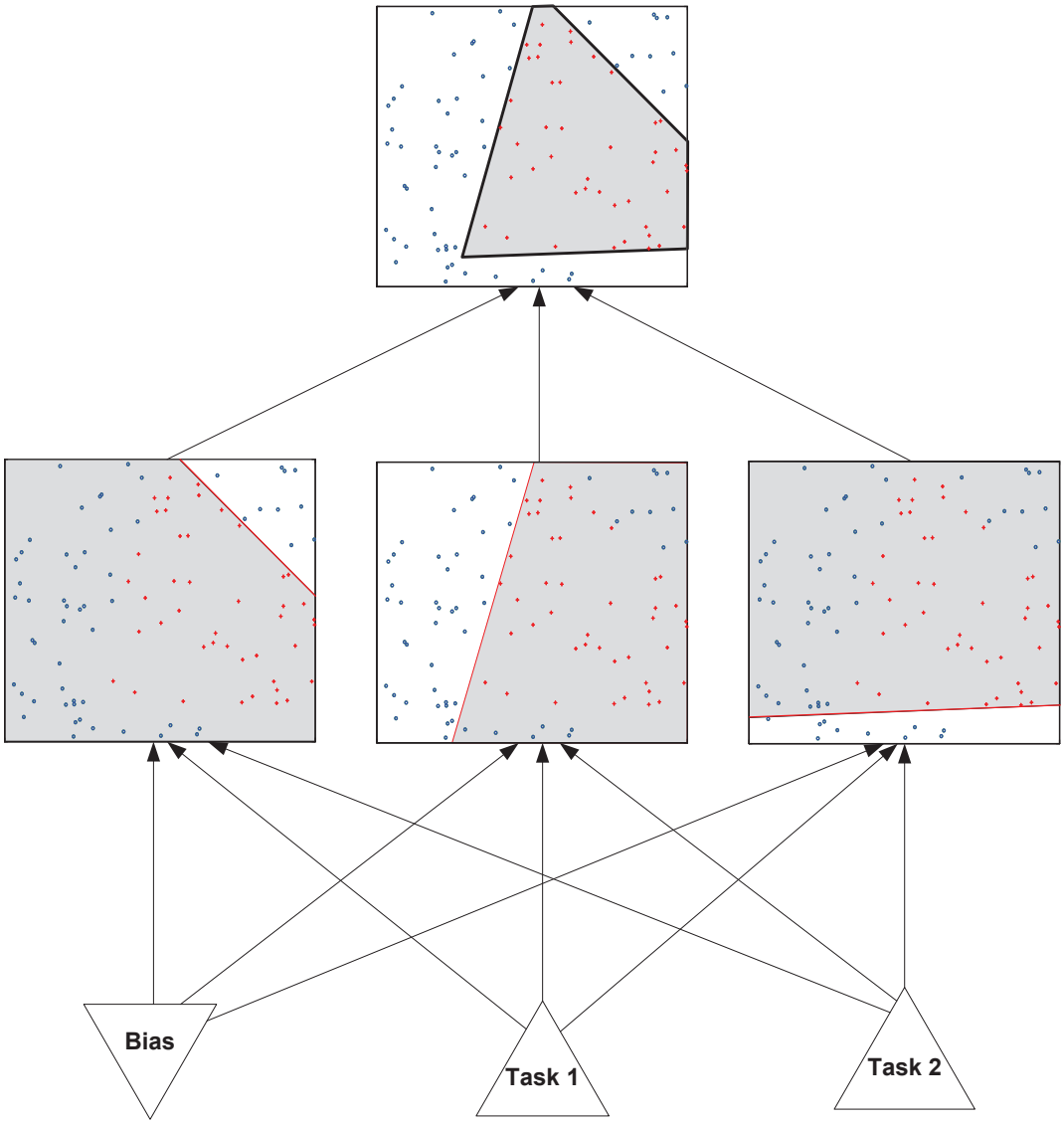


Figure 6.6: *Example of MLP classification.*

MLP has three distinctive characteristics, according to (Haykin, 1999):

1. The model of each neuron in the network includes a nonlinear activation function. The important point to emphasize here is that the nonlinearity is smooth, as opposed to the hard-limiting used in Rosenblatt's perceptron (Rosenblatt, 1959). A commonly used form of nonlinearity that satisfies this requirement is a sigmodial nonlinearity defined by the logistic function:

$$y_j = \frac{1}{1 + exp(-v_j)},$$

where v_j is the induced local field (i.e., the weighted sum of all synaptic inputs plus the bias) of neuron j, and y_j is the output of the neuron. The presence of non-linearities is important because otherwise the input-output relation of the network could be reduced to that of a single-layer perceptron. Moreover, the use of the logistic function is biologically motivated, since it attempts to account for the refractory phase of real neurons.

2. The network contains one or more layers of hidden neurons that are not part of the input or output of the network. These hidden neurons enable the network to learn complex tasks by extracting progressively more meaningful features from the input patterns (vectors).

3. The network exhibits a high degrees of connectivity, determined by the synapses of the network. A change in the connectivity of the network requires a change in the population of synaptic connections or their weights.

The error signal at the output of neuron j at iteration n (i.e., presentation of the nth training example) is defined by

$$e_j(n) = d_j(n) - y_j(n), \quad \textit{neuron } j \textit{ is an output node}, \tag{6.10}$$

We define the instantaneous value of the error energy for neuron j as $\frac{1}{2}e_j^2(n)$. Correspondingly, the instantaneous value $\wp$ of the total error energy is obtained by summing $\frac{1}{2}e_j^2(n)$ over all neurons in the output layer. We may thus write

$$\wp(n) = \frac{1}{2}\sum_{j \in E} e_j^2(n), \tag{6.11}$$

where the set E includes all the neurons in the output layer of the network. Let N denote the total number of patterns contained in the training set. The average squared error energy is obtained by summing $\wp(n)$ over all n and then normalizing with respect to the set size N, as shown by

$$\wp_{av} = \frac{1}{N} \sum_{n=1}^{N} \wp(n) \tag{6.12}$$

For an M-class classification problem, let y_{kj} be the kth output of the network produced in response to the prototype $\vec{x}_j$, as shown by

$$y_{k,j} = F_k(\vec{x}_j), \quad k = 1, 2, ..., M \tag{6.13}$$

where the function $F(\cdot)$ defines the mapping learned by the network from the input to the kth output. Let

$$\begin{aligned} y_j &= [y_{1,j}, y_{2,j}, ..., y_{M,j}] \\ &= [F_1(\vec{x}_j), F_2(\vec{x}_j), ..., F_M(\vec{x}_j)] \\ &= \vec{F}(\vec{x}_j) \end{aligned} \tag{6.14}$$

where $\vec{F}(\cdot)$ is a continuous vector valued function. $\vec{F}(\cdot)$ minimizes the empirical risk functional:

$$R = \frac{1}{2N} \sum_{j=1}^{N} \left\| \vec{d}_j - \vec{F}(\vec{x}_j) \right\|^2, \tag{6.15}$$

where $\vec{d}_j$ is the target output pattern for the prototype $\vec{x}_j$, $\|\cdot\|$ is the Euclidean norm of the enclosed vector, and N is the total number of examples presented to the network in training.

To evaluate the validity of MEB-MLP, we consider a comparison of MLP with MEB-based KT classification against a without KT classification. The MLP classifier employs transferred knowledge $x^{'}$ from primary tasks to secondary task in addition to the original training data for MLP learning. Let $x^{'}$ be the transferred data to the secondary task, then equation 6.13 is extended for MLP+KT as

$$y_{k,j} = F_k([x_j + x^{'}]), \quad k = 1, 2, ..., M. \tag{6.16}$$

6.4.2 MTL Experiments by MEB-based KT-MLP

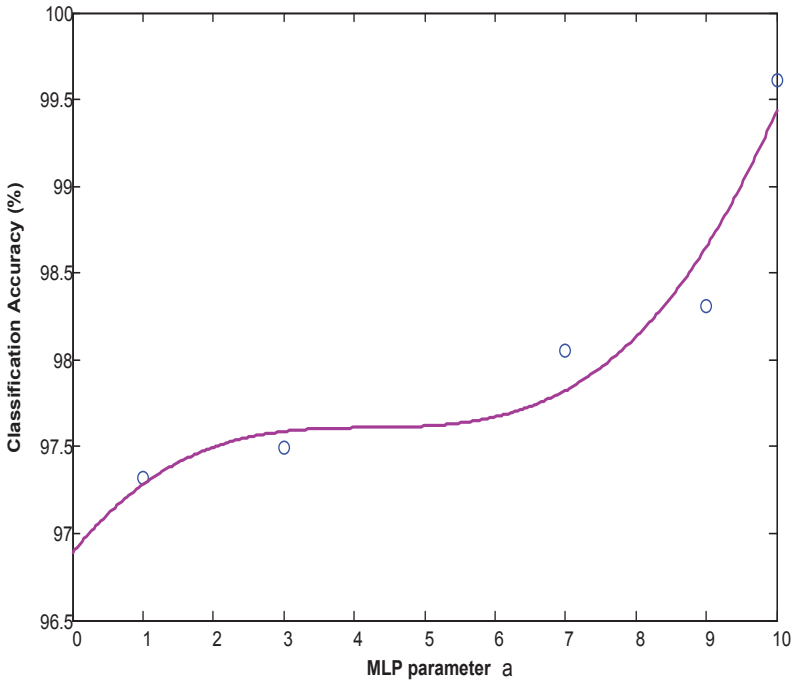
In the MLP learning system, several parameters have to be set. As shown below, those parameters can be divided into two groups: (1) those relates to the MLP classifier, (2) those relates to KT.

[MLP Classifier]
α – Weight decay of MLP.

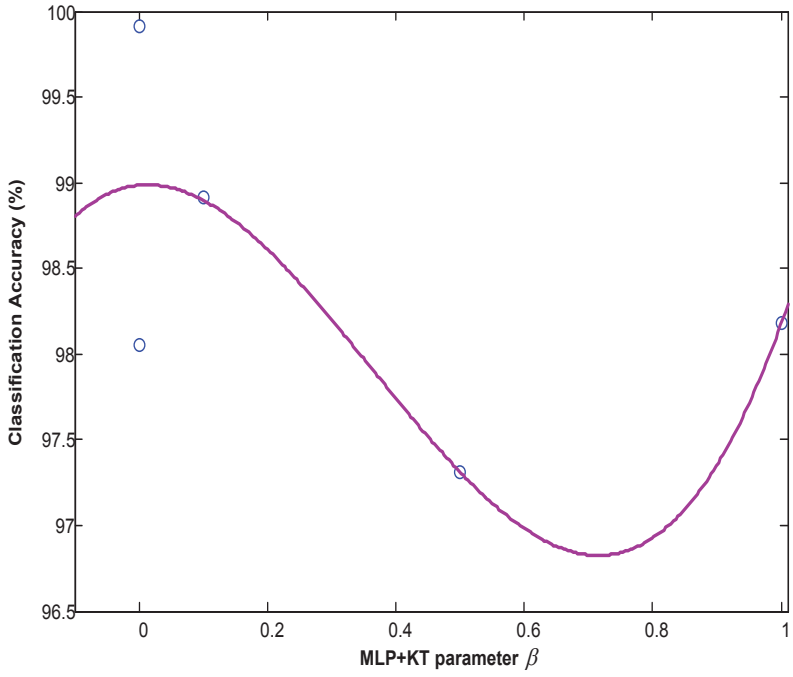
[Knowledge Transfer]
β – It is the control parameter for smoothing MEB, the radius of KT region = the radius of MEB $+\beta$.

Figure 6.7 shows the final classification of MLP classifier for the parameter α and β on the Segmentation dataset. $\alpha = 10$ is the best choice for MLP parameter. We use the same cross validation technique to determine the MLP classifier parameters and β parameter for each dataset. After the decision of parameters, the experimental results on MLP classifier are demonstrated as follows.

Table 6.5 shows final probability of classification accuracy improvement of the pattern recognition tasks based on MLP classifier. As shown in Table 6.5 (a), the best accuracies appear in processing KT amongst two or three pattern recognition tasks on Vowel and Thryoid datasets. In addition, such accuracies decrease, to some extent, to a low level but still approaching to 50%. Afterwards, Table 6.5 (b) reveals acceptable results in comparing accuracies of KT in triple pairs of pattern recognition tasks, such as PI vs GLR, PI vs GER, and PI vs FMA.



(a)



(b)

Figure 6.7: *MEB-based KT-MLP for different model parameters: The classification accuracy for the Segmentation dataset with different values of parameters α and β. The best result is obtained when $\alpha = 10$, and $\beta = 0.1$. (a) Weight decay of MLP α. (b) Smoothing value of KT region β, KT is from Task 2 to Task 1.*

Table 6.5: *Final probability (in percent) of classification accuracy improvement of the pattern recognition tasks based on MLP classifier for the five UCI datasets and the face image dataset. The probability is measured after the completion of 100 runs KT. The three values in each cell are the average probability, average rising accuracy between with-KT and without-KT, and standard deviation in the form of (average) ± (standard deviation).*

(a) UCI Datasets

UCI Datasets	Task 2 → Task 1	Task 1 → Task 2	Task 1 → Task 3
Yeast	**56/100** (0.06% ± 1.01%)	**74/100** (0.24% ± 1.14%)	**44/100** (−0.01% ± 1.01%)
Vowel	**100/100** (3.91% ± 1.11%)	**100/100** (4.07% ± 1.82%)	**100/100** (9.24% ± 3.89%)
Thyroid	**100/100** (0.24% ± 0.11%)	**100/100** (0.63% ± 0.17%)	**100/100** (0.21% ± 0.09%)
Vehicle	**52/100** (0.14% ± 3.90%)	**52/100** (0.06% ± 6.14%)	**48/100** (0.04% ± 5.09%)
Segmentation	**43/100** (−0.01% ± 2.34%)	**53/100** (0.36% ± 9.09%)	**40/100** (−0.29% ± 4.78%)

(b) Face Image Datasets

	PI → GLR	GLR → PI
PI vs GLR	**90/100** (0.88% ± 1.40%)	**0/100** (0.00% ± 0.00%)
	PI → GER	GER → PI
PI vs GER	**92/100** (0.80% ± 1.22%)	**0/100** (0.00% ± 0.00%)
	PI → FMA	FMA → PI
PI vs FMA	**86/100** (0.49% ± 1.22%)	**0/100** (0.00% ± 0.00%)

6.5 Discussion

We have shown the MTL results of both UCI datasets and face image datasets with classifiers of different characteristics. Figure 6.8 demonstrates the performance of 4 UCI datasets by kNN, SVM and MLP classifiers. Surprisingly enough, for the SVM classifier, we obtained a 100% probability of accuracy improvement on the Vowel dataset.

For UCI datasets, the proposed learner independent KT method provides good learning results on both kNN and SVM, the results for MLP are not as good as kNN or SVM. There are two possible reasons for this. First, the proposed KT method extracts a set of raw data from the primary task to the secondary task, kNN, which is a prototype based classifier, and SVM, which is a kernel based classifier, are able to interpolate the transferred knowledge better than a neural network based classifier like MLP. Second, MLP depends on hidden nodes, and the performance of MLP requires a sufficient number of hidden units, whereas some of the UCI datasets could not generate a sufficient quantity of hidden units to partition the feature space. The Vehicle dataset has poor performance on learner independent KT. The reason for this is that the classes of the Vehicle dataset are sparsely distributed, and our pro-

posed MEB-based KT method is not able to extract enough correlation knowledge to improve the learning process.

For the face image dataset, we achieved an overall results of 95 percent of performance improvement on the 3 classifiers, kNN, SVM, and MLP. According to the semantic correlation we mentioned in Chapter 5, the correlated knowledge from PI to GLR, GER, and FMA is more significant that the knowledge in the reverse direction, because PI holds all the information of a person's face, the other 3 tasks (GLR, GER and FMA) are a part of this information (PI).

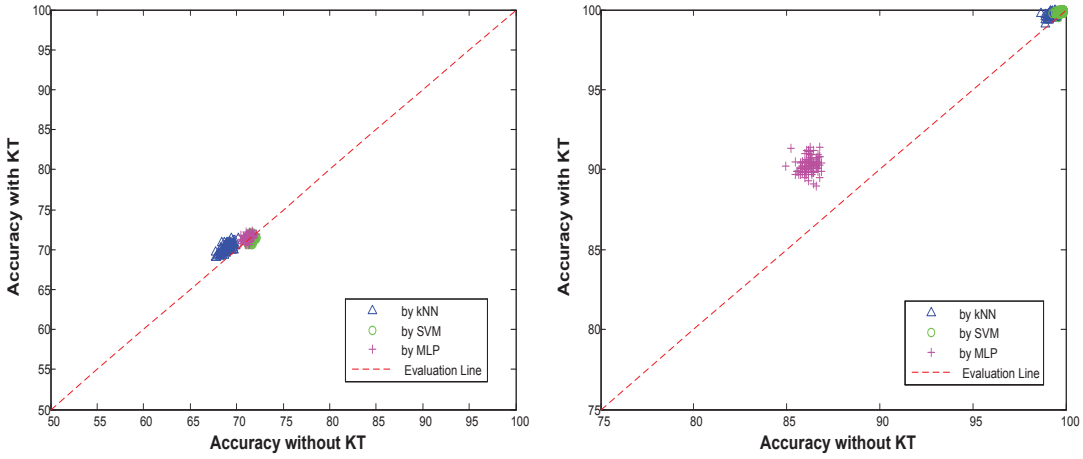
6.6 Summary

Learner independence evaluation on five UCI datasets and real world face image datasets was conducted by the proposed MEB-based KT. Experimentally, the validity of learner independent KT is implemented by 3 different types of classifier, i.e., kNN a prototype based classifier; SVM, a kernel based classifier; and MLP, a neural network based classifier. From the results, it appears that many of the properties hold for the proposed method. We can summarize the main findings as follows:

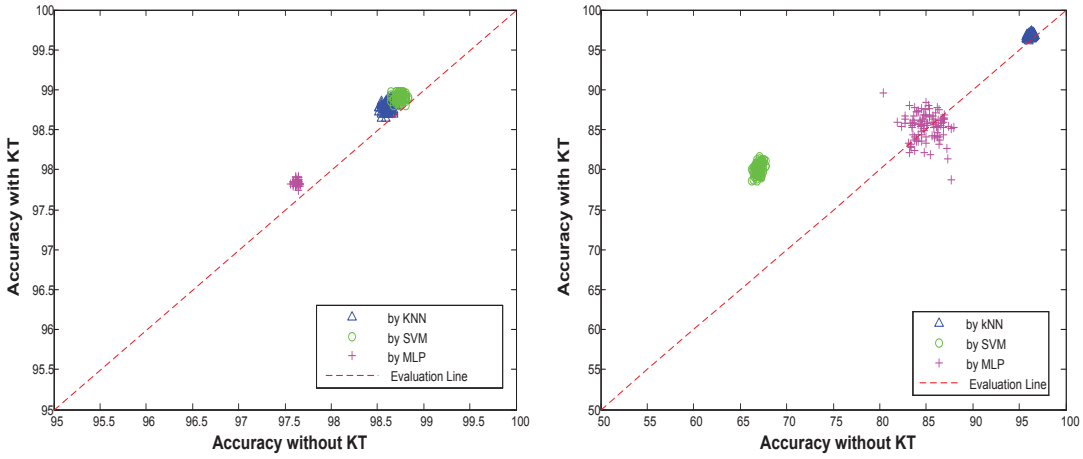
KT effectiveness. The probability of classification accuracy improvement is over 70%, and surprisingly, some tasks can achieve 100 times out of 100 runs where KT successfully enhances classification performance compared with classification without-KT.

KT contributions. The improved accuracy significantly contributes to the secondary task from the primary task via classifiers in various disciplines on most MTL tasks, except the Vehicle dataset.

KT steadibility. The deviation value is not above 10% on the evaluated datasets, which proves the steadibility of the MEB-based KT.



(a) Yeast (b) Vowel



(c) Thyroid (d) Segmentation

Figure 6.8: *Learner independent KT evaluation on UCI datasets by kNN, SVM, and MLP. Points above the dotted line indicate positive KT and those under the line negative KT. (a) Yeast, (b) Vowel, (c) Thyroid, and (d) Segmentation.*

Chapter 7

Conclusions and Future Work

In this chapter, we first summarize our KT method. Secondly, we conclude the strengths and limitations of the proposed method. Finally, we discuss possible future researches based on our KT approach.

7.1 Conclusion on MTL Knowledge Transfer

In this thesis, we studied KT for MTL as a way for sharable knowledge to be extracted from a task, and used in the learning of correlated tasks. While most existing KT methods are learner dependent for MTL, we adopt a learner independent approach and enabled MTL for ordinary classifiers like k-NN, SVM, or MLP.

In the proposed KT, MEB is taken as a size-flexible knowledge representation method, for which multi-resolution domain knowledge mapping is conducted for correlation knowledge extraction from the primary task to the secondary task. $(1 + \delta)r$ MEB expansion and $(1 - \delta)r$ MEB shrinkage both serve to find a maximum subset of sharable samples that have no class confliction. This cast light onto an approach to multi-resolution KT, as correlation is often blurred at varying scales, but is revealed on a smaller scale with higher resolution.

Moreover, MEB is also used as a knowledge container or carrier, such that the contained knowledge is represented as a multi-labelled feature subspace spanned

by shared data from two correlated tasks. Under the smooth assumption of feature space, they can be merged in the same multi-label subspace. At knowledge level, they are sharable by two correlated tasks; at data level, they are a set of data instances spanning the multi-label subspace. Thus they are informative to any classifier which performs learning for a new task.

7.2 Strengths and Limitations of The Proposed Method

The main strengths and limitations of the proposed approach for multi-task learning are summarized in the following.

Adaptability. Since MEB has the characteristics of compactability (optimizes subspace, enclosing the data within one class) and size-flexibility (expanding and shrinking capabilities), the proposed method can adaptively extract the correlation part depending on the size of correlation proportion changing over learning tasks. For example, from a semantic point of view, there is little correlation between glasses recognition and gender recognition, so the KT may result in a small proportion of sharable data. On the other hand, a large correlation exists between glasses recognition and individual identification, the KT may result in a large proportion of sharable data. In general, we are able to evaluate KT effectiveness depending on the semantic relevance understanding from the objective of the learning tasks.

Validity. Experiments conducted on a synthetic dataset, five benchmark datasets (UCI Machine Learning Repository datasets), and a face image dataset show that our approach is effective in exploiting correlated areas and extracting potentially useful data in learning tasks. A learning algorithm with KT, via MEB carriers, outperformed that without KT.

Independence. The proposed KT method has the advantage of learner independence. Experiments are carried out on the proposed KT method together with three well studied learners (e.g., kNN, SVM, and MLP). The results verify that our method can improve accuracy for multi-task learning for all of the tested classifiers.

In the proposed method, we make the assumptions that (1) class labels are known for

every sample with respect to all of the multiple learning tasks, and (2) the feature space is consistent for all learning tasks. In some special cases, two tasks may share the same feature space, but the topic of the learning task is semantically irrelevant, then correlation is considered meaningless. For example, we may have a consistent feature space for a face recognition task and an object recognition task, the correlation extracted from the proposed method may have no significance to improve the learning rate despite the overlap of the different tasks in the feature space.

7.3 Directions for Future Research Work

In this thesis, we focus on the validity of learner independent KT for MTL. For example, MEB-based KT can be instantly applied to any type of learner such as kNN, SVM or MLP. However, technical soundness has not been tested – such as comparing an algorithm with learner-independence with one without learner-independence – this would be worthwhile future work to follow this study.

It is also worth noting that negative transfer happens in the proposed KT in cases that (1) correlation knowledge is not able to be exploited by the classifier, (2) two learning tasks are semantically irrelevant. For example, a face recognition task is not able to be supported by knowledge from a hand writing recognition problem. However, facilitating MTL with unrelated topics is a hot topic in MTL, which will be an interesting direction for future work. Moreover, MTL on related learning tasks over different data domains will also be one of the directions to pursue in future work. Gao et al. (2008), for example, have been working on such KT via multi-model local structure mapping.

References

Abu-Mostafa, Y. S. (1989). Learning from hints in neural networks. *J. Complexity*, *6*(2), 192-198.

Argyriou, A., Maurer, A., & Pontil, M. (2008). An algorithm for transfer learning in a heterogeneous environment. In *Proceedings of the 2008 european conference on machine learning and knowledge discovery in databases - part i* (p. 71 - 85). Antwerp, Belgium: Springer.

Badoiu, M. (2002). Optimal core sets for balls. In *In dimacs workshop on computational geometry.*

Baxter, J. (2000). A model of inductive bias learning. *Journal of Artificial Intelligence Research*, *12*, 149–198.

Ben-David, S., & Schuller, R. (2003). Exploiting task relatedness for multiple task learning. In *Proceedings of computational learning theory (colt).*

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the 5th annual acm workshop on computational learning theory* (pp. 144–152). ACM Press.

Caruana, R. (1997). Multitask learning. *Machine Learning*, *28*, 41-75.

Chan, T. M. (2000). Approximating the diameter, width, smallest enclosing cylinder, and minimum-width annulus. In *Proceedings of the sixteenth annual symposium on computational geometry* (p. 300 - 309). Clear Water Bay, Kowloon, Hong Kong: ACM New York, NY, USA.

Chapelle, O., Scholkopf, B., & Zien, A. (2006). *Semi-supervised learning.* Cambridge, MA: MIT Press.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. In *Machine learning* (pp. 273–297).

Eaton, E. (2006, July 16–20). Multi-resolution learning for knowledge transfer. In . Boston, MA: AAAI Press. ([Doctoral Consortium])

Eaton, E., & desJardins, M. (2006). Knowledge transfer with a multiresolution ensemble of classifiers. In *Icml-06 workshop on structural knowledge transfer for machine learning, june 29, pittsburgh, pa.*

Eaton, E., desJardins, M., & Stevenson, J. (2007). Using multiresolution learning for transfer in image classification. In *aaai07.* AAAI Press.

Evgeniou, T., & Pontil, M. (2004). Regularized multi-task learning. In *Proceedings*

of the tenth acm sigkdd international conference on knowledge discovery and data mining (p. 109-117). ACM.

Gao, J., Fan, W., Jiang, J., & Han, J. (2008, November). *Knowledge Transfer via Multiple Model Local Structure Mapping* (No. 978-1-60558-193-4). ACM.

Good, I. (1980). *Some history of the hierarchical bayesian methodology.* Valencia: Valencia University Press.

Haykin, S. (1999). *Neural networks: A comprehensive foundation.* Upper Saddle River, NJ: Prentice Hall. (2nd edition)

Intrator, N., & Edelman, S. (1998). *Making a low-dimensional representation suitable for diverse tasks.* Norwell, MA, USA: Kluwer Academic Publishers.

Jonathan, B. (1995). Learning internal representations. In *Proceedings of the eighth international conference on computational learning theory.*

Jusman, Y., Sulaiman, S. N., Isa, N. A. M., Yusoff, I. A., Adnan, R., Othman, N. H., et al. (2009). *Capability of new features from ftir spectral of cervical cells for cervical precancerous diagnostic system using mlp networks.* IEEE.

Kasabov, N. (1996). *Foundations of neural networks, fuzzy systems and knowledge engineering.* Cambridge, Massachussets: MIT Press.

Kasabov, N. (2007). *Evolving connectionist systems: The knowledge engineering approach (new edition).* London: Springer Verlag.

Kim, M.-S., Kim, D., & Lee, S.-Y. (2002). Face recognition descriptor using the embedded hmm with the 2nd-order block-specific eigenvectors. *ISO/IEC JTC1/SC21/WG11/M7997, Jeju.*

Kumar, P., Mitchell, J. S. B., & Yildirim, E. A. (2003). Approximate minimum enclosing balls in high dimensions using core-sets. *Journal of Experimental Algorithmics (JEA)*, *8*, 11.

Lawrence, N., & Platt., J. (2004). Learning to learn with the informative vector machine. In *the 21st international conference on machine learning.*

Megiddo, N. (1983). Linear-time algorithms for linear programming in r^3 and related problems. *SIAM Journal on Computing*, *12*(4), 759-776.

Mitchell, T. M. (1980). *The need for biases in learning generalizations* (Tech. Rep.).

Mitchell, T. M. (1997). *Machine learning.* New York: McGraw-Hill.

Ozawa, S., Pang, S., & Kasabov, N. (2006a). Incremental learning of feature space and classifier for on-line pattern recognition. *International Journal of Knowledge based and Intelligent Engineering Systems*, *10*, 57-65.

Ozawa, S., Pang, S., & Kasabov, N. (2006b). Online feature selection for adaptive evolving connectionist systems. *International Journal of Innovative Computing, Information and Control*, *2*(1), 181-192.

Ozawa, S., Pang, S. S., & Kasabov, N. (2008). Incremental learning of chunk data for on-line pattern classification systems. *IEEE Transactions of Neural Networks*, *19*(6), 1061-1074.

Ozawa, S., Roy, A., & Roussinov, D. (2009). A multitask learning model for online pattern recognition. *IEEE Transactions on Neural Networks*, *20*(3), 430-445.

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions*, *PP*(99), 1-1.

Pang, S., Ozawa, S., & Kasabov, N. (2005). Incremental linear discriminant analysis for classification of data streams. *IEEE Transactions of Neural Networks*, *35*(5), 905 - 914.

Rosenblatt, F. (1959). *Principles of neurodynamics*. pub-spartan.

Roy, A., Govil, S., & Miranda, R. (1995). An algorithm to generate radial basis function (rbf)-like nets for classification problems. *Neural Netw.*, *8*(2), 179-202.

Roy, A., Govil, S., & Miranda, R. (1997). A neural network learning theory and a polynomial time rbf algorithm. *IEEE Trans. Neural Netw.*, *8*(6), 1301-1313.

Roy, A., Kim, L. S., & Mukhopadhyay, S. (1993). A polynomial time algorithm for the construction and training of a class of multilayer perceptrons. *Neural Netw*, *6*(4), 535-545.

Roy, A., & Mukhopadhyay, S. (1997). Iterative generation of higher-order nets in polynomial time using linear programming. *IEEE Trans. Neural Netw.*, *8*(2), 402-412.

Silver, D. L. (2000). *Selective transfer of neural network task knowledge*. Phd thesis, University of Western Ontario.

Silver, D. L., & Mercer, R. E. (2002). *Selective functional transfer: Inductive bias from related tasks.*

Silver, D. L., & Poirier, R. (2004). *Sequential consolidation of learned task knowledge.*

Thrun, S. (1996). Is learning the n-th thing any easier than learning the first? In *Advances in neural information processing systems* (p. 640-646). The MIT Press.

Thrun, S., & Pratt, L. (1998). *Learning to learn*. Norwell, MA: Kluwer Academic

Publishers.

Tsang, I. W., Kwok, J. T., & Cheung, P.-M. (2005). Core vector machines: Fast svm training on very large data sets. *The Journal of Machine Learning Research*, *6*, 363 - 392.

Vapnik, V., & Lerner, A. (1963). Pattern recognition using generalized portrait method. *Automation and Remote Control*, *24*.

Welzl, E. (1991). Smallest enclosing disks (balls and ellipsoids). *Results and New Trends in Computer Science*, 359-370.

Xue, Y., Liao, X., & Carin, L. (2007). Multi-task learning for classification with dirichlet process priors. *The Journal of Machine Learning Research*, *8*, 35 - 63.

Yu, K., Schwaighofer, A., Tresp, V., Ma, W.-Y., & Zhang, H. (2003). Collaborative ensemble learning: Combining collaborative and content-based information filtering via hierarchical bayes. In *Proceedings of the 19th conference on uncertainty in artificial intelligence.*