Journal of Machine Learning Research 13 (2012) 691-721 Submitted 3/11; Revised 9/11; Published 3/12

A Case Study on Meta-Generalising: A Gaussian Processes Approach

Grigorios Skolidis G.SKOLIDIS@SMS.ED.AC.UK
Guido Sanguinetti GSANGUIN@INF.ED.AC.UK
School of Informatics
University of Edinburgh
Edinburgh, EH8 9AB, UK

Editors: Sören Sonnenburg, Francis Bach and Cheng Soon Ong

Abstract

We propose a novel model for meta-generalisation, that is, performing prediction on novel tasks based on information from multiple different but related tasks. The model is based on two coupled Gaussian processes with structured covariance function; one model performs predictions by learning a constrained covariance function encapsulating the relations between the various training tasks, while the second model determines the similarity of new tasks to previously seen tasks. We demonstrate empirically on several real and synthetic data sets both the strengths of the approach and its limitations due to the distributional assumptions underpinning it.

Keywords: transfer learning, meta-generalising, multi-task learning, Gaussian processes, mixture of experts

1. Introduction

The central problem of supervised learning is *generalisation*, learning input/ output relations from training data that, when applied to unseen test data, will give good performance (in terms of an appropriate loss function). A common assumption underlying many supervised learning algorithms is that the training and testing data distribution are the same, which allows them to make predictions of future instances of the problem at hand. On the other hand, in the complex world that we live in we are usually faced with unseen but similar problems, situations which human intelligence handles by adaptively taking decisions on the new tasks using knowledge from similar tasks. In this direction, *Transfer learning* (TL) has emerged as a framework to handle situations where there are multiple but related problems to be solved. The term TL is used here in its broader sense, to cover more specific areas of research such as domain adaptation, co-variate shift, sample selection bias, self-taught learning, and multi-task learning. One of the main differences between these subfields of TL lies in the availability of outputs (labels) for input data in the various tasks, no matter if it is a regression or classification problem (Arnold et al., 2007). For example, the situation where labels are available for all tasks is tackled by multi-task learning, which synergistically solves the learning problem in all tasks simultaneously (Caruana, 1997; Bakker and Heskes, 2003; Ando and Zhang, 2005). Domain adaptation (Daumé III and Marcu, 2006; Daumé, 2007; Crammer et al., 2008; Mansour et al., 2009; Pan et al., 2009), co-variate shift (Sugiyama et al., 2007; Storkey and Sugiyama, 2007; Bickel et al., 2009), and sample selection bias (Huang et al., 2007) are settings appropriate for problems where labels are only available for a task that is similar to the task that we wish to make predictions in (target task). Contrary to domain adaptation, and sample selection bias,

self-taught learning (Raina et al., 2007) is a setting where labeled data are available for the target task, but the learning algorithm wishes to also use unlabelled data from a source task to improve performance. In its own right, self-taught learning is distinguishable from semi-supervised learning (Chapelle et al., 2006), where labelled and unlabelled data are assumed to come from the same task. The purpose of all these TL approaches is to enhance the generalisation power of a specific algorithm by leveraging related (but different) knowledge from multiple tasks. In particular, it is generally assumed that at least the input data for the target task will be available *during the learning*, so that a measure of similarity between the training and target tasks can be estimated.

The question that we wish to raise in this work is whether the notion of generalisation can be extended to the level of tasks as a form of *meta-generalisation*. Meta-generalisation is a concept introduced in Baxter (2000), where the author argues whether a transfer learning algorithm can generalise well on totally *unseen* tasks after seeing sufficiently many *source* (or training) tasks. We emphasize that this is much more than a theoretically interesting question. Our motivating example is a strongly applied one: we wish to create an automated diagnosis tool that can accommodate variability among patients, so that, once trained on a sufficient number of patients, it can generalise to new patients. In his work Baxter (2000) derives bounds on the generalisation error of this problem in terms of a generalised VC-dimension parameter, as well as comments that the number of source tasks and examples per task required to ensure good performance on novel tasks has to be sufficiently large. While Baxter (2000) derives an algorithm to select a subset of features to perform multi-task learning based on Neural Networks (NN), his work is more on the theoretical side as no experimental results are presented. Besides that, the model proposed in this work needs to be retrained in case a new target task arrives in order to learn a small number of task dependent parameters.

One way to approach meta-generalising is through domain adaptation, by training a model on the data of the source and the target set of tasks (Ben-David et al., 2007). This type of approach, as well as the model proposed in Baxter (2000), are essentially trained in a transductive way, as the algorithm is able to make predictions only on the test tasks that is trained on, or needs to be retrained in case a new task arrives. Obviously, the performance and the success of domain adaptation algorithms depends strongly on certain assumptions, with most important the similarity between the target and the source distribution (Ben-David et al., 2010). Clearly, if these assumptions are violated then the success of these algorithms is doubtful.

The problem of sampling the space of tasks to make predictions on totally unseen tasks in the inductive setting, which is the exact analog of generalising in the level of tasks, to the best of our knowledge has not been specifically addressed. As we mentioned before, TL is separated into different sub-categories based on the level of supervision on the target task. Hence, multi-task learning can be seen as an *Inductive* TL algorithm since input data and labels are available for all the tasks that we wish to make predictions. On the other end, settings like to Domain adaptation, Covariate shift or Sample selection bias, can be viewed as a form of *Transductive* TL since the algorithm can exploit only the input distribution of the target task they want to make predictions (Arnold et al., 2007). On this basis, meta-generalising can be considered as a form of *Unsupervised* TL, since the learning algorithm does not have any exploitable information about the target tasks during training . Note, that this classification of TL algorithms is different from the one employed in Pan and Yang (2010), where unsupervised TL encapsulates problems like dimensionality reduction, density estimation, or clustering but in situations where multiple tasks are involved, but is in agreement with the taxonomy of TL algorithms introduced in Arnold et al. (2007).

In this paper we investigate the use of coupled Gaussian process models to address this problem. The model uses a multi-class Gaussian process for assigning probabilistically unseen tasks to source tasks (determining task responsibilities), and then uses a multi-task Gaussian process (Bonilla et al., 2008) to perform prediction in individual tasks. Extensive testing on real and simulated data shows the promise of the model, as well as giving insight on the underlying assumptions.

The rest of the paper is organised as follows: in Section 2 we formally define the meta-generalising problem, emphasizing the main assumptions and highlighting the important special case of *fully observed tasks*. In Section 3 and 4 we present our model and the inference methodology used. We present our empirical results in Section 5, and we finish in Section 6 by discussing the merits of our model in the context of the wider literature in transfer learning and meta-generalisation.

2. Meta-generalising

In this section, we formally state the problem of meta-generalising, while we introduce the notation that will be used throughout this paper unless specified otherwise. For simplicity, we concentrate on binary classification problems within each task, while we note that the same formalism applies to regression and multi-class classification problems.

In a meta-generalising scenario the learner is provided with a set of source tasks $\mathcal{T}_S = \{\mathcal{T}_1^s, \ldots, \mathcal{T}_M^s\}$ which are used for training the model; testing is then performed on a set of target tasks $\mathcal{T}_T = \{\mathcal{T}_1^t, \ldots, \mathcal{T}_H^t\}$. Each of the M source tasks will contain a training set of input/ output pairs (x,y), while data from any of the H target tasks are hidden. For later convenience, we will define the whole training set across tasks as a set of triples $T^s = \{x_i^s, y_i^{st}, y_i^{sx}\}_{i=1}^{N^s}$, where $x_i^s \in \mathbb{R}^d$ is the input feature vector, $y_i^{sx} \in \{-1,+1\}$ are the class labels, and $y_i^{st} \in \{1,\ldots,M\}$ is the source task label indicating to which task the input/ output pair pertains, and $N^s = \sum_{j=1}^{M} n_j^s$ is the total number of training pairs where n_j^s is number of data points from the j^{th} source task. Moreover, we will write $X_j^s = \{x_{ij}^s\}_{i=1}^{n_j^s}$ to denote the total item set of the j^{th} source task, while $\mathbf{y}_j^{sx} = \{y_{ij}^{sx}\}_{i=1}^{n_j^s}$ and $\mathbf{y}_j^{st} = \{y_{ij}^{st}\}_{i=1}^{n_j^s}$ will be used to denote all class and task labels from the j^{th} source task. In the rest of the paper subscript j will be used to refer to tasks, and subscript i to data points.

Each of the H target tasks $\mathcal{T}_j^t$ will consist of a set $X_j^t = \{x_{ij}^t\}_{i=1}^{n_j^t}$ of input points, where n_j^t is number of data points from the j^{th} target task and both types of labels are missing. Likewise, the total number of test points will be denoted by $N^t = \sum_{j=1}^{M} n_j^t$. For reasons that will become clear later on, it is further assumed that for each target task data point x_j^t there is information that it comes from the j^{th} target task, but there is no knowledge with which of the source tasks is more similar. Note that each source task training input x_i^s is assigned two types of labels. This implies supervision in both the levels of the tasks and the data, through y^{st} and y^{sx} respectively; task labels y^{st} indicate from which of the source task a specific data point comes from, as a form of *meta-level information*, and class labels y^{sx} indicate to which class inside the task the data point belongs to, as a form of *inter-task information*.

Meta-generalisation, as all machine learning methods, relies on certain assumptions. We concentrate on two basic assumptions; the first one is the *similarity of the distribution* of the target task with at least one of the source tasks, while the second one is the agreement between the labels of the distributions termed as *low-error joint prediction* (Ben-David et al., 2010). Differently from Ben-David et al. (2010), we will define the *low-error joint prediction* between a source and a target task as the error λ_e between their predictive functions f_s and f_t respectively, evaluated at the union

of the source and the target sets $X = X^s \cup X^t$, with $N = N^s + N^t$. Hence, the error λ_e will be given by,

$$\lambda_e = \sum_{i=1}^{N} |f_t(x_i) - f_s(x_i)|,$$

where $x_i \in X$. Intuitively, if the error λ_e is large then there is a disagreement between the labels of the source and target tasks distribution. Also note that, in a multi-task scenario the parameter λ_e can be computed by training two separate models under the same learning framework (e.g., NN, GPs, etc) since labeled data are available for both the source and target task. Thus, the predictive functions of the source and target task can be estimated separately and λ_e can also be used as an empirical measure of the relatedness of the two tasks. Conversely, in the scenarios of meta-generalising and domain adaptation one has to *assume* that the error λ_e will be low, since labels are available only for the source tasks. If one of these assumptions is not valid, then meta-generalisation can not be expected to guarantee success.

We now give a formal definition of meta-generalising.

Definition 1 *Given a set of source tasks $\mathcal{T}_S$ and a set of target tasks $\mathcal{T}_T$, meta-generalising is an inductive inference method that aims at making predictions on the set of target tasks by sampling the space of source tasks .*

We further define two possible scenarios: in the *fully observed tasks* case, we assume that the similarity of the distribution assumption is perfectly met, so that the data generating distribution of the target task is the same as that of one of the source tasks (but we do not know which one). This assumption is relaxed in the *partially observed tasks* scenario, where we still assume similarity of the distribution but we do not necessarily have identity.

The meta-generalising setting implies that there is hierarchical structure in the problem. The data of each task are on the base level and the distribution of the tasks is on the meta level. Hence, it is intuitive that mechanisms are required to

1. Model the distribution of the data of each task, and the distribution of the source tasks (correlation between tasks).

2. Infer the level of correlation between the target task and the source tasks.

The first prerequisite leads us to multi-task learning, as many approaches offer mechanisms to model both the data and the task distribution (Bakker and Heskes, 2003; Yu et al., 2005; Ando and Zhang, 2005; Xue et al., 2007; Argyriou et al., 2008; Bonilla et al., 2008; Daumé III, 2009). Following the multi-task route, informally speaking, the second prerequisite can be translated as the problem of which of the M outputs of the multi-task classifier to select to make predictions for the target task. In some cases, task-descriptor features may be available, giving a direct measure of task similarity. In this work, we are interested in the general case where no reliable task descriptor features are available; we will then learn similarities between tasks through a *distribution matching* pursuit.

Another way of approaching the problem of meta-generalisation is through the framework of *mixtures of experts* (ME) (Jacobs et al., 1991; Waterhouse, 1997), under which a bigger learning problem is broken down to smaller subproblems that are handled by individual experts. The underlying assumption of this framework is that the data are generated by different processes (Waterhouse, 1997, Ch. 2), an assumption that can also be made in the multi-task setting about the

data generating mechanism of each task; under the ME framework each expert is used to model the data generating process of each subproblem. These experts are then combined through a gating network that models the responsibilities of the experts on each data partition. Hence, attacking the meta-generalisation problem through the ME framework can be seen as an unsupervised alternative method to that problem, that does not use the information about the origins of each task (the source task labels) but instead allows the algorithm to automatically infer the data partitions and the regions of expertise of each expert. Therefore the ME approach is in direct connection to multi-task learning and meta-generalisation in which cases the experts are equivalent to the tasks, and this framework could be used as a rough lower bound on the performance of a multi-task classifier. Note though that in principle it would be desirable to be able to automatically infer the number of experts as in Rasmussen and Ghahramani (2001) which can be seen as a similar mechanism of finding cluster of tasks, in contrast with the method of ME with GPs in Tresp (2000) where the number of experts had to be known *a priori*.

3. A Model for Meta-generalisation

Having identified the nature of the problem, we now propose a model for meta-generalising. The model builds upon the multi-task learning framework of Bonilla et al. (2008) which is able to capture the dependencies between the data and the tasks. In addition, we employ a classifier over the tasks to learn the task labels (from which task each data point comes from). Both of those two learning mechanisms, multi-task setting and classification of the tasks, are modeled by Gaussian Processes (GPs), which are coupled by sharing a common hyper-prior. In the rest of this section, we first give a short introduction to GPs and we review multi-task learning with GPs of Bonilla et al. (2008), we then present the model for meta-generalising, and finally we describe how to make predictions on new tasks.

3.1 Multi-task Learning with Gaussian Processes

Gaussian processes (Rasmussen and Williams, 2005) provide a flexible modelling framework for supervised learning which has become increasingly popular in recent years. A Gaussian Process is a probability distribution over functions f, where the joint distribution of function evaluations over a finite set of inputs is a multivariate Gaussian distribution. At core of the GP prediction is the *covariance function* or *kernel*, parameterised by θ^x, that models the output covariance at different pairs of input points, and in essence acts as a measure of similarity between different input locations. In order for a covariance function to be valid it has to be positive semidefinite, and has to satisfy Mercer's theorem (Rasmussen and Williams, 2005).

In a multi-task scenario the interest lies in learning M related functions $\mathbf{f}_j,\ j = 1,\ldots,M$, from training data $x_{ij},\ y_{ij},\ \ i = 1,\ldots,n_j$, with $x \in\ \mathbb{R}^d$, and $n_1 + \ldots + n_M = N$. In the following of this section, data points from task j will be denoted by $X_j = [x_{1j},\ldots,x_{n_j j}]$ and $\mathbf{X} = [X_1,\ldots,X_M]$ will be used to denote the set of all data points. Focussing on a regression problem for simplicity, the noise model will be given by

$$y_{ij} = f_j(x_{ij}) + \varepsilon_j, \text{ with } \varepsilon_j \sim \mathcal{N}(0,\sigma_j^2), \tag{1}$$

where y_{ij} (x_{ij}) denotes the i^{th} output (input) of the j^{th} task. We note that each input point has M function values associated with it (one per task); this *complete set of responses* will rarely be observed in

practice, but function values corresponding to unobserved values can easily be marginalised using the consistency of GPs

The multi-task model of Bonilla et al. (2008), which has been known in the geo-statistics community as the "*Intrinsic Model of Coregionalization*" (IMC) (Cressie, 1993), can be elegantly recovered from the theory of matrix variate distributions (Gupta and Nagar, 2000). Define the vector $\mathbf{f}$ by stacking the columns of $\mathbf{F} = [\mathbf{f}_1 \dots \mathbf{f}_M]$ into a single vector, $\mathbf{f} = \text{vec}(\mathbf{F})$, where $\mathbf{f}_j \in \mathbb{R}^{N\times 1}$ is the column vector of all latent functions evaluations of task j. Then the *probability density function* of matrix $\mathbf{F}$ will be given by:

$$(2\pi)^{-\frac{1}{2}NM}|\mathbf{K}^t|^{-\frac{1}{2}N}|\mathbf{K}^x|^{-\frac{1}{2}M}\exp\left\{-\frac{1}{2}\text{trace}\left(\left(\mathbf{K}^t\right)^{-1}\mathbf{F}\left(\mathbf{K}^x\right)^{-1}\mathbf{F}^T\right)\right\}, \tag{2}$$

where $\mathbf{K}^t \in \mathbb{R}^{M\times M}$ and $\mathbf{K}^x \in \mathbb{R}^{N\times N}$ (Gupta and Nagar, 2000). This configuration implies that the matrix $\mathbf{K}^t$ models the correlations between the vectors $\mathbf{f}_j$, that is, the tasks in the multi-task view, and $\mathbf{K}^x$ models the correlations between each element of vectors $\mathbf{f}_j$. In the GP framework, this correlation between function evaluations at different input points is captured by the covariance function. Then, by using some matrix algebra involving the vec and Kronecker operator, Equation (2) can be written in the form Bonilla et al. (2008) proposed,

$$p(\mathbf{f}|\mathbf{X}) = \mathcal{GP}(0, \mathbf{K}^t \otimes \mathbf{K}^x).$$

Employing this type of prior for the latent functions $\mathbf{f}$ the noise model for the regression problem stated in equation (1) becomes, $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{f}, \mathbf{D}\otimes\mathbf{I})$, where $\mathbf{D} \in \mathbf{R}^{M\times M}$ is diagonal with $D_{jj} = \sigma_j^2$ and $\mathbf{I} \in \mathbb{R}^{N\times N}$ is the identity matrix.

The key element of this formulation is the task covariance matrix $\mathbf{K}^t$ which reflects the task correlations. For example, if $\mathbf{K}^t$ was fixed to the identity matrix, then all tasks would be independent but they would still share the same hyperparameters of the covariance function. Of course, one of the main goals of multi-task learning is to learn these task dependencies. Bonilla et al. (2008) approached this problem by parameterizing the task covariance matrix, with parameters θ^t, always retaining positive definite restrictions, and treating these parameters as hyperparameters to be learned. Positive definite guarantees were achieved, by parameterizing a lower triangular matrix L to employ the Cholesky factorization $\mathbf{K}^t = LL^T$. Most importantly, parameters related to the data covariance function or the task covariance matrix can be learned in the standard GP formulation, by maximizing the marginal likelihood $p(\mathbf{y}|\mathbf{X}) = \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{X})\text{d}\mathbf{f}$.

3.2 Model

In this section we describe the Coupled Multi-Task Multi-Class (CMTMC) model we propose for meta-generalisation. The objectives of the model are first to model the dependencies between the tasks, and second to assign unseen tasks to source tasks by finding task similarities. The first objective is met through the Multi-task part of the model, while the second is achieved through the Multi-class classifier. Figure 1 shows the graphical model of the CMTMC classifier. In this subsection we use x, y^t, and y^x to refer to x^s, y^{st}, and y^{sx} to keep the notation light, since in the learning phase only source tasks are involved. Therefore, notation introduced in Section 3.1 applies here. Moreover, from Section 2 we have that $y^t \in \{1,\dots,M\}$ and $y^x \in \{-1,+1\}$ as the task and class labels respectively. Since both class and task prediction are effectively classification models, we choose the probit and multinomial probit models as noise models respectively. Following Albert

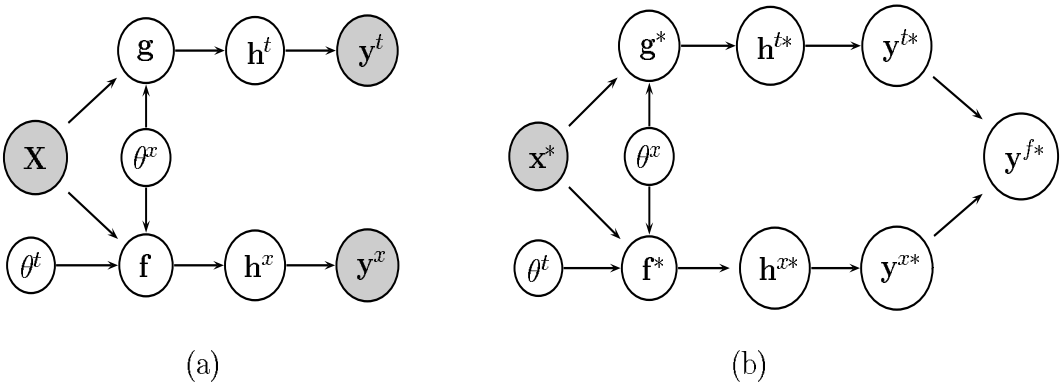


Figure 1: Coupled Multi-Task Multi-Class (CMTMC) model. Variables **f** and **g** are the two sets of GPs for the multi-task and multi-class classifiers respectively, whereas variables $\mathbf{h}^x$ and $\mathbf{h}^t$ denote the auxiliary variables of the two classifiers; (a) graphical representation of the training phase, (b) graphical representation of Meta-generalising.

and Chib (1993), we define two sets of auxiliary variables $\mathbf{h}^t = \text{vec}(\mathbf{H}^t)$, and $\mathbf{h}^x = \text{vec}(\mathbf{H}^x)$, which as shown later on enables the multinomial and the binary probit model respectively. For later convenience, we will be using $\mathbf{h}^t_j$ and $\mathbf{h}^t_n$ to denote the j^{th} column and n^{th} row of matrix $\mathbf{H}^t$.

Figure 1 shows that there are two directed channels of variables. The upper channel, with variables $C^t = \{\mathbf{g}, \mathbf{h}^t, \mathbf{y}^t\}$, is responsible for learning the task labels, thus from which task each data point comes from, while the lower channel, with variables $C^x = \{\mathbf{f}, \mathbf{h}^x, \mathbf{y}^x\}$, learns to classify the data points inside every task and to find task correlations, through the standard multi-task classifier.

Thus, there are two sets of Gaussian Processes. The first one is responsible for the classification over the tasks $\mathbf{g}|\mathbf{X}, \theta^x \sim \mathcal{GP}(0, \mathbf{I} \otimes \mathbf{K}^x)$, where $\mathbf{g} = \text{vec}(\mathbf{G})$, $\mathbf{G} = [\mathbf{g}_1, \ldots, \mathbf{g}_M]$, and $\mathbf{g}_j \in \mathbb{R}^{N \times 1}$. The second one is responsible for the multi-task classification $\mathbf{f}|\mathbf{X}, \theta^x, \theta^t \sim \mathcal{GP}(0, \mathbf{K}^t \otimes \mathbf{K}^x)$, where as stated before variables θ^x and θ^t are used to denote the hyperparameters of the data covariance function and task matrix respectively. As in the multi-class case we will have that $\mathbf{f} = \text{vec}(\mathbf{F})$, where $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_M]$ and $\mathbf{f}_j \in \mathbb{R}^{N \times 1}$. In the rest of the paper we will write $\mathbf{K}^x$ to denote the covariance matrix between all data points $\mathbf{X}$, unless specified otherwise. Moreover, $\mathbf{I}$ and $\mathbf{K}^t$ will be $M \times M$, where the identity matrix in the multi-class case implies independence between the classes, thus $\mathbf{g}_j|\mathbf{X}, \theta^x \sim \mathcal{GP}(0, \mathbf{K}^x)$. The key objective is to learn M functions $\mathbf{g}_j$ for the multi-class classifier and M related functions $\mathbf{f}_j$ for the multi-task classifier.

Note that the data covariance matrix $\mathbf{K}^x$ is shared by both sets of processes **g** and **f**. This is graphically illustrated by the fact that the node of hyperparameters θ^x is connected to both latent functions; thus, the multi-class and the multi-task classifier share the same hyperparameter space for θ^x. The multi-class classifier is restricted to have the same covariance function across the classes in contrast with the standard model for multi-class classification with GPs, which in principle allows you to use different covariance functions across classes. In fact, the CMTMC model could be decoupled into two separate classifiers with different sets of hyperparameters θ^x between the two processes **f** and **g**. Seemingly, this decoupling would result in a more flexible model, but preliminary experiments with both models, the CMTMC and the decoupled model, has shown that this

restriction does not affect the performance. In contrast, it reduces dramatically the computational cost since the hyperparameters of the data covariance function need to be estimated only one time.

The probit model is enabled in both channels by a standardized normal noise model over the auxiliary variables, $h^t_{ij}|g_{ij} \sim \mathcal{N}(g_{ij},1)$, and $h^x_i|f_i \sim \mathcal{N}(f_i,1)$ (Albert and Chib, 1993; Csató et al., 2000; Girolami and Rogers, 2006; Skolidis and Sanguinetti, 2011). The relationship between outputs y^t and y^x and auxiliary variables h^t, and h^x is deterministic and will be given by:

$$y^t_i = j \text{ if } h^t_{ji} = \max_{1\leq k\leq M}\{h^t_{ki}\},$$

$$p(y^x_i|h^x_i) = \begin{cases} \delta(h^x_i)\delta(y^x_i) & \text{if } y^x_i = +1 \\ \delta(-h^x_i)\delta(-y^x_i) & \text{if } y^x_i = -1 \end{cases},$$

where δ is one if its argument is positive and zero otherwise, which completes the specification of the model.

3.2.1 INFERENCE

Classification problems imply non-Gaussian noise models, which make inference intractable. To address this intractability, we adopt a variational approximate treatment to the problem, as it is computationally more efficient than sampling-based methods while retaining a reasonable accuracy in empirically approximating posterior marginals.[1] For a comprehensive comparison between these approximations for GP multi-class classification, and on the multinomial probit model the interested reader in referred to Girolami and Rogers (2006). The dependencies of the random variables $\Theta = \{\mathbf{g},\mathbf{h}^t,\mathbf{f},\mathbf{h}^x\}$ are depicted graphically in Figure 1.a and are summarized in the joint likelihood of the CMTMC model as:

$$p(\mathbf{y}^t,\mathbf{y}^x,\Theta|\theta^x,\theta^t,\mathbf{X}) = p(\mathbf{y}^t|\mathbf{h}^t)p(\mathbf{h}^t|\mathbf{g})p(\mathbf{g}|\theta^x,\mathbf{X})p(\mathbf{y}^x|\mathbf{h}^x)p(\mathbf{h}^x|\mathbf{f})p(\mathbf{f}|\theta^x,\theta^t,\mathbf{X}).$$

Variational methods approach this problem by approximating the joint posterior of the latent variables Θ within a family of tractable distributions; in our case, we will approximate the joint posterior as a factored distribution $p(\Theta|\mathbf{y}^t,\mathbf{y}^x,\mathbf{X},\theta^t,\theta^x) \approx Q(\Theta) = \prod_{i=1} Q(\Theta_i) = Q(\mathbf{g})Q(\mathbf{h}^t)Q(\mathbf{f})Q(\mathbf{h}^x)$. Minimizing the Kullback-Leibler divergence between the approximating and the true distribution is equivalent to maximizing the following lower bound on the marginal likelihood

$$\log p(\mathbf{y}^t,\mathbf{y}^x|\mathbf{X},\theta^x,\theta^t) \geq \int Q(\Theta)\log\frac{p(\mathbf{y}^t,\mathbf{y}^x,\Theta|\mathbf{X},\theta^x,\theta^t)}{Q(\Theta)}\mathrm{d}\Theta, \tag{3}$$

which is found by applying Jensen's inequality (MacKay, 2003). Standard results show that the distributions that maximize the lower bound are given by

$$Q(\Theta_i) = \frac{\exp(\mathbb{E}_{Q(\Theta\setminus\Theta_i)}\{\log\, p(\mathbf{y}^t,\mathbf{y}^x,\Theta|\mathbf{X},\theta^t,\theta^x)\})}{\int\exp(\mathbb{E}_{Q(\Theta\setminus\Theta_i)}\{\log\, p(\mathbf{y}^t,\mathbf{y}^x,\Theta|\mathbf{X},\theta^t,\theta^x)\})\mathrm{d}\Theta_i}$$

where $Q(\Theta\setminus\Theta_i)$ denotes the factorized distribution with the i^{th} component removed. Inference and learning are performed in a variational EM algorithm: the E-step computes the variational posteriors on the variables Θ, and the M-step optimizes the hyperparameters θ^t,θ^x given the expectations

1. Another setting for approximate inference producing comparable results with the Variational approach that could have been employed is the EP approximation (Opper and Winther, 2000; Minka, 2001; Rasmussen and Williams, 2005); this has also been extended to the multi-class classification scenario in Girolami and Zhong (2007).

computed in the previous step. At each (E or M) iteration the variational lower bound, $\mathcal{L}(Q)$ (given in Appendix B Equation (15)), provably increases (or at worst remains unchanged), and these two steps are repeated until convergence.[2] We now briefly summarize the calculations needed to perform the E and M steps. The pseudo-algorithm of the training of the CMTMC model is given in Algorithm 3.2.1. We omit any details and emphasize only the occurrence of the special form covariance function we employ; fuller details can be found in Appendices A, and B.

E-step. The approximate posteriors for the multi-class classifier will be given by,

$$Q(\mathbf{g}) = \prod_{j=1}^{M} \mathcal{N}_{\mathbf{g}_j}(\tilde{\mathbf{g}}_j, \boldsymbol{\Sigma}^g), \tag{4}$$

$$Q(\mathbf{h}^t) = \prod_{n=1}^{N} \mathcal{N}^{y_n^t}_{\mathbf{h}_n^t}(\tilde{\mathbf{g}}_n, \mathbf{I}), \tag{5}$$

where $\boldsymbol{\Sigma}^g = \left(\mathbf{I} + (\mathbf{K}^x)^{-1}\right)^{-1} = \mathbf{K}^x (\mathbf{I} + \mathbf{K}^x)^{-1}$, $\tilde{\mathbf{g}}_j = \boldsymbol{\Sigma}^g \tilde{\mathbf{h}}_j^t$, *and* $\mathcal{N}^{y_n^t}_{\mathbf{h}_n^t}(\tilde{\mathbf{g}}_n, \mathbf{I})$ *denotes an M-dimensional Gaussian distribution truncated such that* j^{th} *dimension has the largest value if* $y_n^t = j$. *In the lower channel, the approximate posteriors for the multi-task classifier will be given by,*

$$Q(\mathbf{f}) = \mathcal{N}_{\mathbf{f}}(\tilde{\mathbf{f}}, \boldsymbol{\Sigma}^f), \tag{6}$$

$$Q(\mathbf{h}^x) = \prod_{i=1}^{NM} \left(\tilde{f}_i + y_i^x \frac{\mathcal{N}_{\tilde{f}_i}(0,1)}{\Phi(y_i^x \tilde{f}_i)} \right), \tag{7}$$

where $\tilde{\mathbf{f}} = \boldsymbol{\Sigma}^f \tilde{\mathbf{h}}^x$, *and* $\boldsymbol{\Sigma}^f = \mathbf{K}^t \otimes \mathbf{K}^x (\mathbf{I} + \mathbf{K}^t \otimes \mathbf{K}^x)^{-1}$ *and the tilde notation in the above random variables denotes posterior expectation, that is,* $\tilde{t}(\alpha) = \mathbb{E}_{Q(\alpha)}\{t(\alpha)\}$*; more details can be found in Appendix A.*

M-step. The M-step optimises the lower bound with respect to the hyperparameters θ^x and θ^t. This is performed by gradient descent; computation of the gradients of the lower bound given in Equation (3) are somewhat intricate and are given in Appendix B.

Algorithm 1 CMTMC model - Training

1: **Inputs** : X_j^s, $\mathbf{y}_j^{sx}$, $\mathbf{y}_j^{st}$ for $j = 1, \ldots, M$
2: Sample parameters $\mathbf{g}, \mathbf{h}^t, \mathbf{f}, \mathbf{h}^x$ from prior
3: Initialise hyper-parameters θ^x, θ^t
4: **repeat**
5: **E-step**
6: Compute $Q(\mathbf{g})$ and $Q(\mathbf{h}^t)$ for MC-classifier, Equations (4),(5)
7: Compute $Q(\mathbf{f})$ and $Q(\mathbf{h}^x)$ for MT-classifier, Equations (6),(7)
8: **M-step**
9: Optimize hyperparameters θ^x, θ^t, Equations (16), (16)
10: Compute Lower-bound on log-marginal likelihood, Equation (15)
11: **until** convergence

2. In practice, estimation of the convergence of the EM algorithm was inferred when the increase between iterations was zero or smaller than a very small constant.

3.3 Prediction on Novel Tasks

While in the previous section we described how to train the model on training data from the source tasks, we now describe how to perform predictions on unseen target tasks. We adopt a mixture of experts type approach; in these networks, multiple outputs are combined and weighted according to the responsibilities they have on a certain prediction task. In a similar manner, the multi-task classifier of the CMTMC model can be seen as a multi-output predictor, and the classifier over the task labels (multi-class) can be used to infer the responsibilities of the outputs of the multi-task classifier, since it produces posterior probabilities of task memberships. Then predictions on novel tasks are computed according to

$$p(y^{f*} = +1|x^*, \mathbf{X}, \mathbf{y}^t, \mathbf{y}^x) = \sum_{j=1}^{M} p(y_j^{x*} = +1|x^*, y^{t*}, \mathbf{X}, \mathbf{y}^x) p(y_j^{t*}|x^*, \mathbf{X}, \mathbf{y}^t), \tag{8}$$

where $p(y_j^{x*} = +1|x^*, y^{t*}, \mathbf{X}, \mathbf{y}^x) = p(y_j^{x*} = +1|x^*, \mathbf{X}, \mathbf{y}^x)$ is the posterior of the j^{th} task belonging to class "+1" from the multi-task classifier, and $p(y_j^{t*}|x^*, \mathbf{X}, \mathbf{y}^t)$ is the posterior of x^* coming from the j^{th} task, or the test point task responsibility from the multi-class classifier. A graphical representation of this process is given in Figure 1.b, where it is shown that nodes y^{t*}, and y^{x*} are combined to give the final predictions y^{f*}.

However, the meta-generalisation scenario presents some additional challenges which are not found in classical mixture of experts models. In many cases, a target task consists of a *batch* of input points, and the simple fact that they all come from the same task contains valuable information about the correlations between the associated outputs. Another closely related issue is that of the correlation between the target task and the source tasks. In many multi-task problems it is a usual phenomenon to observe groups of highly correlated tasks (e.g., Figure 3.b), while other times tasks are correlated but in a more random fashion (e.g., Figure 6.b, 7.b). As we will see in the experimental sections, this can have important consequences in terms of predictive accuracy, and in terms of choosing an appropriate prediction model.

In the following, we present two distinct scenarios for inferring the task responsibilities. Given a target task with n^t data points $\mathbf{x}^{t*} = \{x_1^{t*}, x_2^{t*}, \ldots, x_{n^t}^{t*}\}$, in the first scenario we treat each data point from the target task individually to infer its task responsibilities, which we will refer to as *Point to Point Gating* (P2PGat). This approach neglects the information that all target points come from the same task, and as we will see in the experimental section, is more appropriate when inter-task correlations are weaker. In the second scenario we wish to combine the information from all n^t test points to infer the overall task responsibilities for the target task, which we will refer to as *Batch* predictions.

3.3.1 POINT TO POINT GATING

Given a new input point which lacks both class and target labels, the CMTMC model combines the predictions of a multi-task classifier using task responsibilities obtained from the multi-class classifier channel. Thus, two sets of quantities need to be computed. The first set are the posterior

probabilities of the M outputs $p(y_j^{x*} = +1|x^*, \mathbf{X}, \mathbf{y}^x)$ of the multi-task classifier, as

$$p(y_j^{x*} = +1|x^*, \mathbf{X}, \mathbf{y}^x) = \int p(y_j^{x*} = 1|h^{x*}) p(h^{x*}|x^*, \mathbf{X}, \mathbf{y}^x) \mathrm{d}h^{x*},$$
$$\equiv \int_0^{+\infty} \mathcal{N}_{h_j^{x*}}(\lambda_j^*, \upsilon_j^{*2}) \mathrm{d}h^{x*} = \Phi\left(\frac{\lambda_j^*}{\upsilon_j^*}\right) \tag{9}$$

where we have used that $\upsilon_j^{*2} = 1 + k_{jj}^t k_{x^*x^*}^x - \left(\mathbf{k}_j^t \otimes \mathbf{k}_{\mathbf{x},x_*}^x\right)^T \left(\mathbf{I} + \mathbf{K}^t \otimes \mathbf{K}^x\right)^{-1} \left(\mathbf{k}_j^t \otimes \mathbf{k}_{\mathbf{x},x_*}^x\right)$, and $\lambda_j^* = \mathbf{k}_j^t \otimes \mathbf{k}_{\mathbf{x},x^*}^x \left(\mathbf{I} + \mathbf{K}^t \otimes \mathbf{K}^x\right)^{-1} \tilde{\mathbf{h}}^x$. Additionally, $\mathbf{k}_j^t$, k_{jj}^t are used to denote the j^{th} column and the jj^{th} element of $\mathbf{K}^t$ respectively, $\mathbf{k}_{\mathbf{x},x_*}^x$ is used to denote the covariance vector between $\mathbf{X}$ and x^*, and Φ is the probit function.

The second set of quantities are the task responsibilities which are computed from Girolami and Rogers (2006)

$$p(y^{t*} = k|x^*, \mathbf{X}, \mathbf{y}^t) = \int p(y^{t*} = k|h^{t*}) p(h^{t*}|x^*, \mathbf{X}, \mathbf{y}^t) \mathrm{d}h^{t*}$$
$$\equiv \int_{-\infty}^{+\infty} \mathcal{N}_{h_k^{t*}}(\mu_k^*, \nu_k^*) \prod_{m \neq k} \int_{-\infty}^{h_k^{t*}} \mathcal{N}_{h_m^{t*}}(\mu_m^*, \nu_m^*) \, \mathrm{d}h_m^{t*} \, \mathrm{d}h_k^{t*}, \tag{10}$$

which can be evaluated using numerical integration as:

$$p(y^{t*} = k|x^*, \mathbf{X}, \mathbf{y}^t) = \mathbb{E}_{p(u)} \left\{ \prod_{j \neq k} \Phi\left(\frac{1}{\nu_j^*} \left[u\nu_k^* + \mu_k^* - \mu_j^*\right]\right) \right\}, \tag{11}$$

where $u \sim \mathcal{N}_u(0,1)$, $\nu_m^* = 1 + k_{x^*,x^*}^x - \mathbf{k}_{\mathbf{x},x^*}^{x^T} \left(\mathbf{I} + \mathbf{K}^x\right)^{-1} \mathbf{k}_{\mathbf{x},x^*}^x$, and $\mu_m^* = \mathbf{k}_{\mathbf{x},x^*}^{x^T} \left(\mathbf{I} + \mathbf{K}^x\right)^{-1} \tilde{\mathbf{h}}_m^t$.

In the P2PGat scenario, the novel input points are not assumed to share a common task label. Therefore, class prediction is performed straightforwardly on every new input by inserting the posterior probabilities obtained in Equations (9,10) in the gating network given by Equation (8).

3.3.2 BATCH

In a Bayesian way using all test points $\mathbf{x}^{t*}$ to infer the overall task responsibility is performed by replacing the univariate distributions from Equation (10) with the appropriate multivariate. As a result the second integral of Equation (10) becomes the multivariate cumulative distribution function $\int_{-\infty}^{\mathbf{h}_k^{t*}} \mathcal{N}_{\mathbf{h}_m^{t*}}(\mathbf{M}_m^{g*}, \boldsymbol{\Upsilon}^*) \, \mathrm{d}\mathbf{h}_m^{t*}$. Specifically the mean and the variance of the auxiliary variables $\mathbf{h}_m^{t*}$ on the batch of test points $\mathbf{x}^*$ will be given by:

$$\mathbf{M}_m^{g*} = \mathbb{E}[\mathbf{h}_m^{t*}|\mathbf{x}^*] = \mathbf{K}_{\mathbf{x},\mathbf{x}^*}^{x^T} \left(\mathbf{I} + \mathbf{K}_{\mathbf{x},\mathbf{x}}^x\right)^{-1} \tilde{\mathbf{h}}_m^t \tag{12}$$

$$\boldsymbol{\Upsilon}^* = \mathrm{cov}[\mathbf{h}_m^{t*}|\mathbf{x}^*] = \mathbf{I} + \mathbf{K}_{\mathbf{x}^*,\mathbf{x}^*}^x - \mathbf{K}_{\mathbf{x},\mathbf{x}^*}^{x^T} \left(\mathbf{I} + \mathbf{K}_{\mathbf{x},\mathbf{x}}^x\right)^{-1} \mathbf{K}_{\mathbf{x},\mathbf{x}^*}^x, \tag{13}$$

where $\mathbf{K}_{\mathbf{x},\mathbf{x}^*}^{x^T}$ is the $N \times n^t$ covariance matrix of all training points $\mathbf{X}$, and all test task data points $\mathbf{x}^*$, and $\mathbf{K}_{\mathbf{x}^*,\mathbf{x}^*}^x$ is the $n^t \times n^t$ full covariance matrix of $\mathbf{x}^*$. Equations (12) and (13), indicate that inferring the tasks responsibilities on a set of points depends not only on the correlations between the test points and the train points but also on the correlations between the test points themselves.

Algorithm 2 CMTMC model - Meta-generalising

Inputs : $\mathbf{x}^{t*} = [x_1^{t*}, \ldots, x_{n^t}^{t*}]$, $Q(\mathbf{g})$, $Q(\mathbf{h}^t)$, $Q(\mathbf{f})$, $Q(\mathbf{h}^x)$, $\mathbf{X}$
for $i = 1$ to n^t **do**
  for $j = 1$ to M **do**
    Compute MC posterior probabilities $p(y_{ij}^{t*} = j | x_i^{t*}, \mathbf{X}, \mathbf{y}^t)$, Equation (11)
    Compute MT posterior probabilities $p(y_{ij}^{x*} = +1 | x_i^{t*}, \mathbf{X}, \mathbf{y}^x)$, Equation (9)
  end for
end for
P2PGat predictions
for $i = 1$ to n^t **do**
  Compute $p(y^{f*} = +1 | x^*, \mathbf{X}, \mathbf{y}^t, \mathbf{y}^x)$, Equation (8) based on steps 4 and 5
end for
BATCH predictions
for $j = 1$ to M **do**
    Compute overall task posterior probabilities $p(\mathbf{y}^{t*} = k | \mathbf{x}^*, \mathbf{X}, \mathbf{y}^t)$, Equation (14)
end for
for $i = 1$ to n^t **do**
  Compute $p(y^{f*} = +1 | x^*, \mathbf{X}, \mathbf{y}^t, \mathbf{y}^x)$, Equation (8) based on steps 5 and 14
end for

On the other hand, truncated multivariate Gaussian distributions are hard to deal with, and usually approximations are applied (Deak, 1980; Genz, 1992; Gassmann et al., 2002). The dimensions of the multivariate distribution function in the batch prediction problem depend on the number of data points n_* of the target task, which can be several thousands depending the application. To the best of our knowledge no method can tackle very high dimensional c.d.f. , and even approximations can become extremely computationally intensive when n_* is more than a few dozens (these estimations would be carried out within the inner loop of a VBEM algorithm, which would obviously further aggravate the problem). A solution to this problem is to assume that data points from the test task are i.i.d. from the unknown data generating distribution, and approximate it by:

$$p(\mathbf{y}^{t*} = k | \mathbf{x}^*, \mathbf{X}, \mathbf{y}^t) \approx \frac{\prod_{i=1}^{n^*} p(y_i^{t*} = k | x_i^*, \mathbf{X}, \mathbf{y}^t)}{\sum_{m=1}^{M} \prod_{j=1}^{n^*} p(y_j^{t*} = m | x_j^*, \mathbf{X}, \mathbf{y}^t)}, \tag{14}$$

where $p(y_i^{t*} = k | x_i^*, \mathbf{X}, \mathbf{y}^t)$ are the task responsibilities computed individually for each test point. We will adopt this approximation in the experimental section for computational reasons; calculations using the full covariances in Equation (13) are unfeasible with more than 100 points (test or training). While this approximation may appear crude, we experimented extensively in medium-scale problems using a reduced rank approximation for $\boldsymbol{\Upsilon}^*$ (capturing up to 90% of the total variance), but this did not appear to yield significant empirical advantages justifying the substantial computational costs. Note though that although the i.i.d. approximation misses the correlations between the test samples, it still uses information from all test points to produce overall test task class posterior probabilities.

The pseudo-algorithm for the stage of Meta-generalisation for both types of predictions, P2PGat and Batch, is given in algorithm 3.3.1.

4. Experiments

This section aims at providing insights into the workings of our meta-generalising model through empirical evidence. Experiments are presented for both the fully observed and partially observed task scenarios described in Section 2, and in both cases we investigate both the P2P gating and the Batch mode of predictions on new tasks. The fully observed tasks case, considered in Section 4.1, investigates the situation where data generating distribution of the target task is actually the same as that of one of the source tasks. In this case all available tasks are used in the training phase, but in the testing phase the model has no information from which of the source task the target task comes from. The second set of experiments, described in Section 4.2, considers the case of the partially observed tasks. In this case the data generating distribution of the target task does not match the distribution of one of the source tasks, so that the set of source tasks is strictly a subset of the set of all tasks. Training is performed on the source tasks, and testing on the totally unseen target tasks. While both scenarios are plausible applications of meta-generalising, Section 4.2 gives more insight into the connections between the correlation structure of the tasks and the task prediction mechanism on totally unseen tasks.

Five different data sets are considered in the experiments. The first two data sets are artificially generated to demonstrate the strengths and the limitations of the method; the first one satisfies the assumptions of the model, and the second one, which is only considered in Section 4.1, is in conflict with them. The third data set is a character classification problem between commonly confused handwritten letters. The fourth data set is an automated diagnosis problem: annotated heartbeats from ECG recordings are used to discriminate normal from arrhythmic beats, and each patient is considered as a task. The last data set, which is considered only in the second set of experiments, is a landmine detection problem. More details are given in each section separately. We present results for different training set sizes, and for each training size experiments are repeated 25 times by randomly selecting the data points used for training from each task. Furthermore, in both scenarios three types of outputs are considered from the CMTMC model; the batch written as "BatchMCAppr", the P2P gating written as "P2PMCGat", and the "MAP" estimate which simply selects the output of the multi-task classifier that has the highest posterior, something that is usually considered in classifier fusion techniques (Kuncheva, 2002). As our method essentially relies on the covariance structure between tasks, two types of baseline comparisons are possible: in the worst case, results should not be worse than completely ignoring the task structure and pooling together all training data. We refer to this baseline as Pool. In the best case, our method should not be statistically better than a method which leverages the same covariance structure and has access to all the task label information, for example, a standard multi-task learning approach. We refer to this best-case scenario as MTL; we compare with this only in the fully observed task scenario, as in the partially observed case the meta-generalising results are generally quite far from this best case.

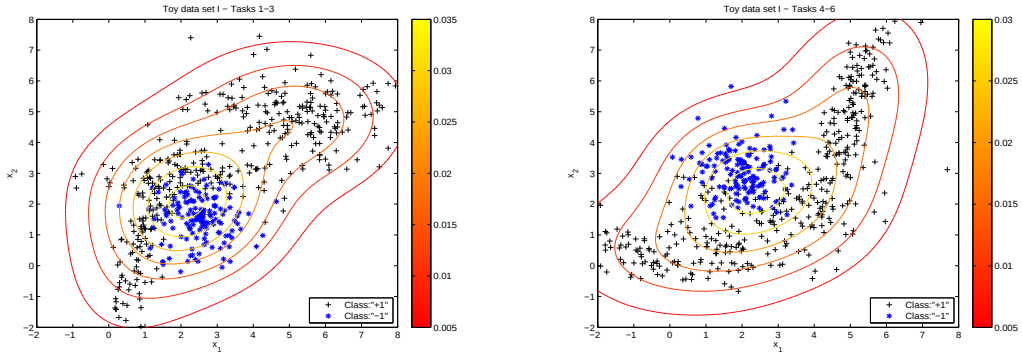
All methods are compared in terms of the area under the precision-recall curve, also known as the *Average Precision* (AP) (Davis and Goadrich, 2006). Simulation results were processed based on the work of Brodersen et al. (2010), that provides a smooth estimate of the precision-recall curve;[3] an equivalent performance measure that could have been used is the Area Under the Curve (AUC) (Hanley and Mcneil, 1982), which is also appropriate for imbalanced data sets. Note that simulation results follow the same pattern with both measures. In all experiments the task covariance matrix K^t

3. Code downloaded from: `http://people.inf.ethz.ch/bkay/downloads`.

was parameterized as a correlation matrix (Rebonato and Jäckel, 2000), with unit diagonal, while the data covariance function K^x is set specifically for each data set depending the application.

4.1 Fully Observed Tasks

In this scenario, the data distribution of the target task is the same as that of (at least) one of the source tasks. This guarantees that the similarity of distribution assumption is met, however, as we'll see in the case of Toy data *II*, the low joint prediction error assumption is not automatically satisfied. Obviously, the actual input data will be different, due to the stochasticity of the data generating process. Intuitively, the success of the model depends strongly on whether the model will be able to infer correctly from which of the source tasks the target task actually comes from.



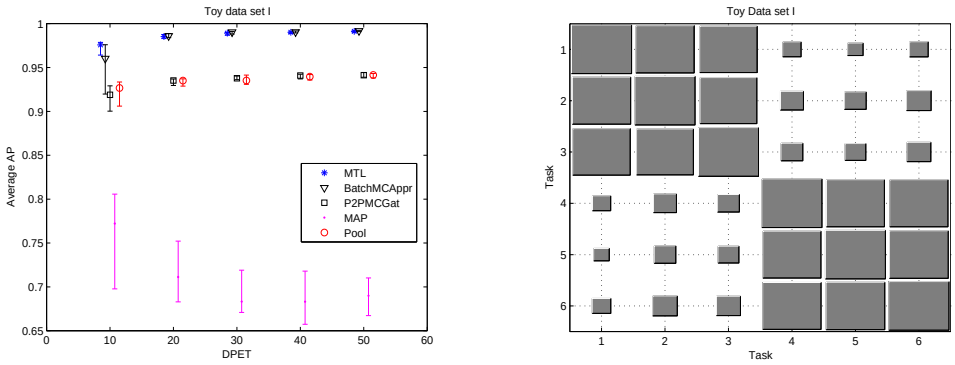
(a) (b)

Figure 2: Toy data set I distribution; (a) scatter plot and density for the first cluster of tasks (1-3), (b) scatter plot and density for the second cluster of tasks (4-6).

4.1.1 TOY DATA SET *I*

The first toy data set is comprised of six binary classification tasks. This toy problem was previously used in Liu et al. (2009) in the context of semi-supervised multi-task learning. Data for the first three tasks are generated from a mixture of two partially overlapping Gaussian distributions, and similarly for the remaining three tasks. Hence, the six tasks cluster in two groups; for each task 600 data points were generated, which were equally divided between the two classes. The scatter plots of the two clusters are shown in Figures 2.a and 2.b.

This data set is ideal for demonstrating the concept of the meta-generalising for three reasons. First of all the assumptions of the model are satisfied. Secondly, the tasks group in two clusters. The third reason is that the densities of the clusters though similar are not exactly the same; this is illustrated in Figures 2.a and 2.b, which shows the contour plot of the densities of the two clusters. We use an Automatic Relevance Determination (ARD) data covariance function, which employs a different characteristic length scale for each feature, and is able to identify which features are more relevant for classification (Rasmussen and Williams, 2005).



(a) (b)

Figure 3: Toy data set I classification Results; (a) Average AP over the 6 tasks, (b) Hinton Diagram of the task covariance matrix of the CMTMC model computed by averaging over the 25 repetitions with 50 data points per task.

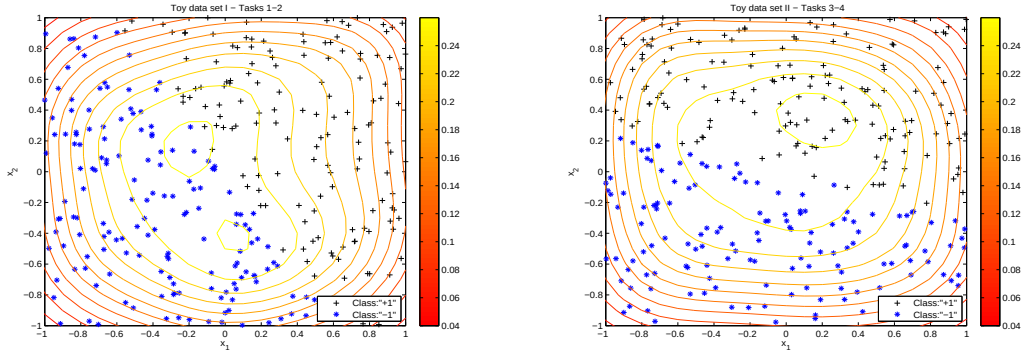
Classification results are presented in Figure 3.a; the Y axis is the AP, and the X axis is the number of data points from each task (DPET) used for training. The results show that, in this toy problem, the Batch mode performs similarly to the ideal MTL case, although it has a high variance for the case of 10 DPET. The P2PGat and Pooling method perform approximately 10% worse than the Batch, while the MAP estimate gives roughly 20% less than the Batch. Moreover, Figure 3.b shown the Hinton diagram[4] (Hinton, 1989) of the task covariance matrix of the CMTMC model which accurately recovers the structure of the tasks.

4.1.2 TOY DATA SET *II*

The second toy data set consists of four tasks which group into two clusters. The scatter plot as well as the density of the two clusters are shown in Figures 4.a and 4.b, for the first and second cluster respectively. The main feature of this data set, evident visually from Figure 4, is the similarity of the data generating distribution for the two tasks. While the densities are peaked in different locations, without class labels the tasks are almost identical, meaning that the multi-class classifier cannot learn to discriminate between the two tasks. As in the previous example, each task consisted of 600 data points equally divided between the two classes, and we used the ARD covariance function.

Figure 5.a shows the results the different methods produced. As expected, the Batch mode fails to correctly identify the task responsibilities; as a result, it gives a lower average AP than the MTL, a difference which does not decrease with the number of DPET, indicating statistical inconsistency. This is reinforced by the Hinton diagram of $\mathbf{K}^t$ in Figure 5.b, where it fails to identify the clusters of the tasks. Even though this difference is small it is significant for this easy problem, where the MTL algorithm performs close to 100%. Additionally, the P2PGat, the Pooling, and the MAP estimates perform better that the Batch, but they also fail to reach the performance of MTL.

4. The Hinton diagram is a graphical representation of the values in a data matrix; here, it is used to display the correlations between the tasks.

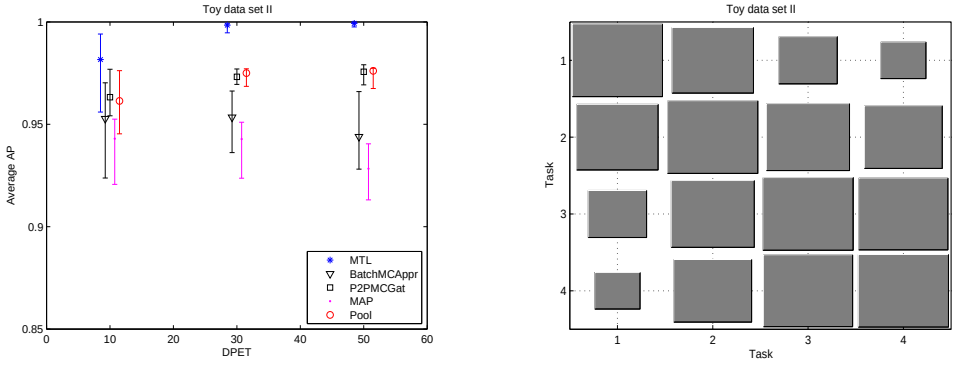


(a) (b)

Figure 4: Toy data set II distribution;(a) scatter plot and density for the first cluster of tasks(1-2), (b) scatter plot and density for the second cluster of task(3-4).

4.1.3 CHARACTER CLASSIFICATION

In this data set the task is to learn to classify between commonly confused handwritten letters, which is included in the "Transfer learning Toolkit" of Berkeley University available at `http://multitask.cs.berkeley.edu/`. This data set is comprised of eight binary classification tasks. The characters that are used and the number of samples are given in Table 1. Each sample is a 16×8 image, which results into a binary 128 feature vector. The covariance function that is employed for this data set is the *Radial Basis Function* (RBF).



(a) (b)

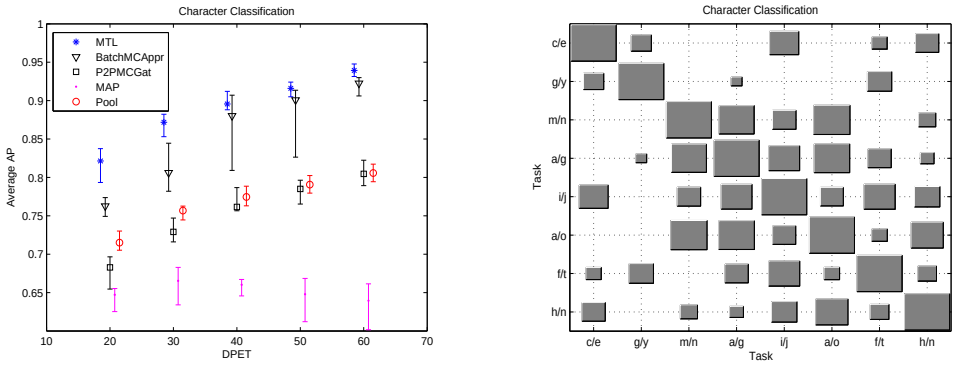
Figure 5: Toy data set II classification Results; (a) Average AP over the 4 tasks, (b) Hinton Diagram of the task covariance matrix.

The classification results for this data set are presented in Figure 6.a. The Batch method follows closely the ideal MTL performance, and outperforms the P2PGat, Pooling, and the MAP methods

Task	1	2	3	4	5	6	7	8
Letter	**c**	**g**	**m**	**a**	**i**	**a**	**f**	**h**
Number of data	2017	2460	1596	4016	4895	4016	918	858
Letter	**e**	**y**	**n**	**g**	**j**	**o**	**t**	**n**
Number of data	4928	1218	5004	2460	188	3880	2131	5004

Table 1: Description of the Character data set; each column is a task showing the two letters as well as the corresponding number of examples per character.

(although there is significant variability for small numbers of labeled data per task). Figure 6.b shows the Hinton diagram of the task covariance matrix, which indicates a more random structure between the tasks, but finds that some tasks are more correlated than others, for example 'a/g' with 'a/o', and 'i/j' with 'f/t'. It should be noted though, that in this data set the "low-error joint prediction" assumption is partially violated since there is label disagreement between tasks 'a/g' and 'g/y', where the 'g' letter belongs to class "+1" in task 'a/g' and to "-1" in task 'g/y'. This does not seem to have any adverse effect on the performance of the model, presumably as the difference between letters 'a' and 'y' is sufficient to unambiguously assign the target task to the correct source task.



(a) (b)

Figure 6: Character Classification Results; (a) Average AP over the 8 tasks, (b) Hinton Diagram of the task covariance matrix.

4.1.4 ARRHYTHMIA CLASSIFICATION

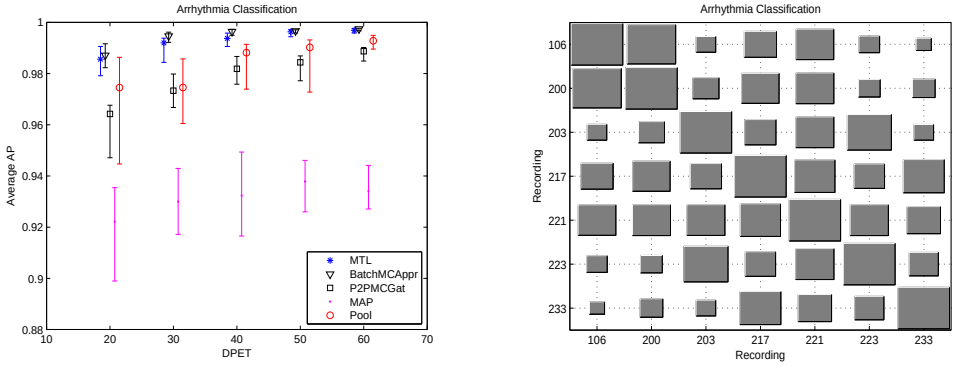
The arrhythmia data set consists of seven ECG recordings from different patients, which were acquired from the MIT-BIH Arrhythmia database (Goldberger et al., 2000). Each recording corresponds to a large number of heart beats, which is summarized in Table 2. Each patient is treated as a separate task, and the goal is to classify each heart beat into two classes, normal or premature ventricular contraction (PVC) arrhythmic beats. The same problem was considered in Skolidis et al.

(2008) using single task GP classifiers. Each recording was sampled at 360Hz, and annotation provided by the database was used to separate the beats before any preprocessing. Each beat segment, consisting of 360 data points (one minute), was transformed into the frequency domain using a Fast Fourier Transform with a Hanning window. Only the first ten harmonics are used as features for classifying heart beats, as most of the information of the signal is contained in these harmonics.

Recording ID	106	200	203	217	221	223	233
Total number of data	2021	2567	2970	406	2349	2417	3053
Number of Normal heart beats	1503	1740	2526	244	1954	1955	2224
Number of PVC heart beats	518	827	444	162	395	462	829

Table 2: Description of the Arrhythmia data set.

Figure 7.a shows the average AP over the seven tasks. On average, the Batch method performs better than the P2PMCGat, the MAP, and the Pool, while it presents a small advantage compared to MTL. Interestingly, the MAP approach is consistently worse than other methods, a situation that will be reversed in the partially observed tasks scenario. As in the character classification problem the task covariance matrix $\mathbf{K}^t$, shown in Figure 7.b, demonstrates that there are correlations between the tasks but in more random way.



(a) (b)

Figure 7: Arrhythmia Classification Results; (a) Average AP over the 7 tasks, (b) Hinton Diagram of the task covariance matrix.

4.1.5 OBSERVATIONS

This set of experiments has demonstrated the effectiveness of the CMTMC model in situations where the data distribution of the target task comes from one of the source tasks. Several observations are made:

1. In the fully observed tasks scenario, the space of tasks has been sampled sufficiently (by definition). In this case the Batch mode should theoretically be the best method, since all data

points are needed to produce an accurate estimate of the density of the target task. This is empirically confirmed in our investigation, as Batch closely approaches the MTL results in all cases.

2. If the "low-error joint prediction" assumption is violated, then meta-generalising becomes a very hard problem, possible unsolvable. The performance on the second toy example was not particularly bad, since all methods achieved higher that 90% in terms of AP, but none of methods reached the performance of the MTL algorithm, and the performance did not appreciably improve when more training data were provided, indicating statistical inconsistency. This effect could be dramatically increased if for example the classes between the clusters were anti-correlated, so that similar data generating distributions could be potentially associated with opposite predictions. Note though that if discriminative task descriptor features are available then this problem can be overcome, because augmenting the feature space would result in a different mapping of the latent function f.

3. If the model assumptions are met, the correlation structure of the tasks does not have a strong influence on the predictions, since the Batch mode outperformed the P2PGat gating and MAP estimate in all experiments. As we will see, this will be a crucial difference between the fully and partially observed tasks scenario.

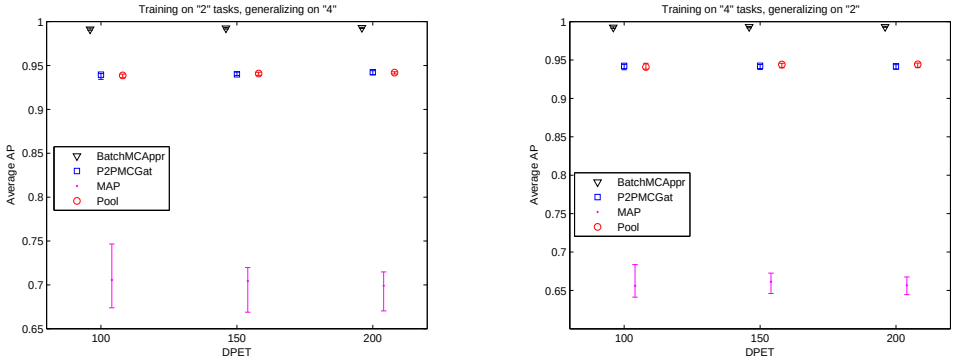
4.2 Partially Observed Tasks

We now consider the harder problem of making predictions on completely unseen tasks. In this case, *a priori* we have no guarantee that any of the underlying modelling assumptions (similarity of distribution and low-error joint prediction) may hold. However, in some situations it is not unrealistic to assume that inter-task correlations will be structured, for example by the presence of *clusters* of similar tasks. These clusters may be evident from the experimental design of the problem (as in the case of the landmine data set discussed below), or may become evident from the training phase on the source tasks, if the learned task covariance matrix exhibits a strong block structure.

We are not aware of other methods that has a distribution matching mechanism to perform predictions on totally unseen tasks. Therefore, in this section we will only compare the different inference mechanisms of the CMTMC model (Batch and P2PGat) with a GP model trained by pooling all data together and with the MAP combination of classifiers.

4.2.1 TOY DATA SET *I*

We consider the toy data set that was used in Section 4.1.1 consisting of two clusters of tasks; in this section, training tasks are selected by randomly selecting equal number of tasks from each cluster. The challenge for the model is to correctly classify the task, given the similarity of the task distributions between the two clusters (see Figure 2). While it could be argued that, as the tasks in each cluster have the same data generating distribution, this example is very close to the fully observed case scenario (and it certainly is if we consider the underlying tasks to be two rather than six), it is still a useful illustrative example as a limiting case where assumptions are perfectly met. Experimental results are presented for two and four training tasks in Figures 8.a and 8.b respectively. Naturally, as this data set is designed to match our modelling assumptions, the Batch method outperforms all other methods; it is interesting however that the method successfully detects from which cluster of tasks the unseen target task comes from even for relatively small training set



(a) (b)

Figure 8: Average AP on the unseen tasks of Toy data set *I*; (a) training on 2 tasks generalising on 4, (b) training on 4 tasks generalising on 2.

sizes. Comparing the performance of the Toy data set *I* in the fully and partially observed cases, in Figures 3 and 8 respectively, reveals that the same levels of AUC are achieved in both experimental setups, indicating that the task classification GP is highly confident of the correct result.

4.2.2 LANDMINE DETECTION

The landmine detection data set consists of images measured with airborne radar systems, and the goal is to predict landmines or clutter (Xue et al., 2007). Data are collected from 19 landmine fields, which are considered as subtasks, and each point is represented by a nine-dimensional feature vector. Tasks 1-10 correspond to regions that are relatively highly foliated while tasks 11-19 correspond to regions that are bare earth or desert. Figure 9 shows the number of data points from each task and each class, which indicates that this data set is highly imbalanced in favor of the Clutter ('-1') class. The experimental setup suggests the presence of two clusters of tasks corresponding to the

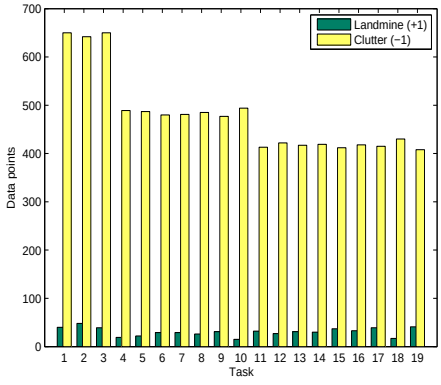
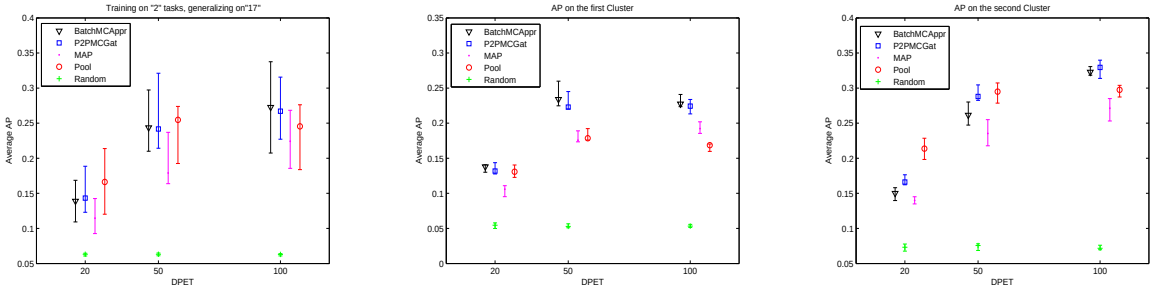


Figure 9: Landmine detection data distribution.

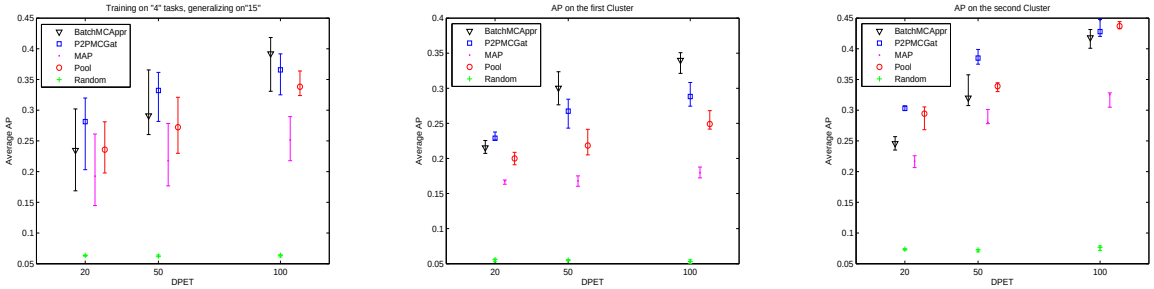
geomorphology of the region the observations come from; this is confirmed by our preliminary investigation (not shown), as well as from previously published results on this data set by Xue et al. (2007) and Liu et al. (2009). Thus, in this data set training tasks are set by randomly selecting equal number of tasks from the first cluster, tasks 1-10, and from the second cluster, tasks 11-19. Experiments are presented for two, four, and eight training tasks. The data covariance function that is used for this data set is the ARD.

Figures 10.a, 11.a, and 12.a shows the mean AP on the 17, 15, and 11 unseen target tasks for each partition respectively. Due to the high imbalance between the classes (Landmine-Clutter) the achieved AP of all methods is very low. Therefore, in this data set we also present the AP of a random predictor which clearly shows the improvement of each method considered. Note that in terms of AUC the results obtained in this work are consistent with previous studies in this data set (Xue et al., 2007; Liu et al., 2009), which are presented in Appendix C Figure 14 for completeness. Moreover, it is noticed that there are large overlapping error bars between all methods. Large error bars give evidence that there might be two levels of performance. Therefore, for each partition we provide the average AP for each cluster separately; subfigures (b) from Figures 10, 11, and 12 show the average AP for the first cluster, and subfigures (c) for the second cluster. Measuring the AP in each cluster separately gives significantly smaller error bars, and reveals interesting structures in the problem. Specifically, the performance on the second cluster is always better than on the first cluster by a considerable margin. Moreover, comparing the methods on each cluster separately we see that the Batch method outperformed the pooling and the P2PGat in most of the cases, particularly in the first cluster where the advantages become very significant as we increase the number of tasks/ DPETs. The correlation structure within the second cluster is looser, immplying a weaker applicability of our modelling assumptions. However it should be pointed out that this is a substantially harder pattern recognition task compared to the toy data set considered above. For example, Liu et al. (2009) that investigated the application of semi-supervised MTL on this data set achieved a best performance of 78% AUC; the CMTMC (which relies on the more flexible GP framework for MTL) achieves an average AUC above 76% on totally unseen tasks having trained on *only* 8 source tasks with 100 DPET (see Figure 14).



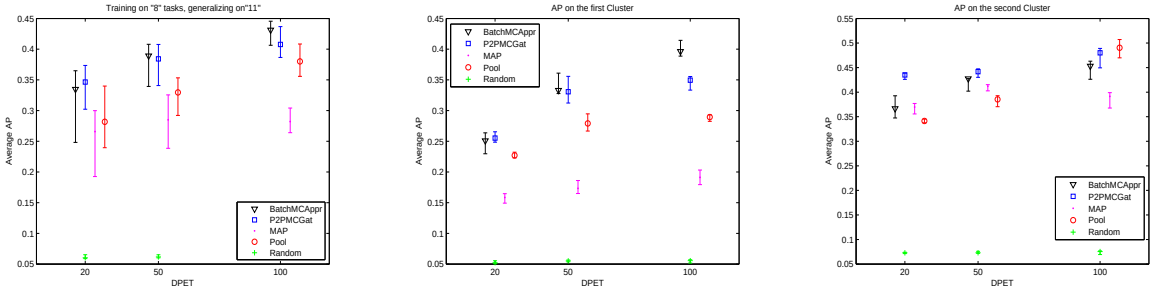
(a) (b) (c)

Figure 10: AP on the 17 unseen tasks of Landmine data set; training on 2 tasks, generalising on 17; (a) AP over 17 tasks, (b) AP over 9 tasks of the first cluster, (c) AP over 8 tasks of the second cluster.



(a) (b) (c)

Figure 11: Average AP on the 15 unseen tasks of Landmine data set; training on 4 tasks, generalising on 15; (a) Overall AP over 15 tasks, (b) Average AP over 8 tasks of the first cluster, (c) Average AP over 7 tasks of the second cluster.



(a) (b) (c)

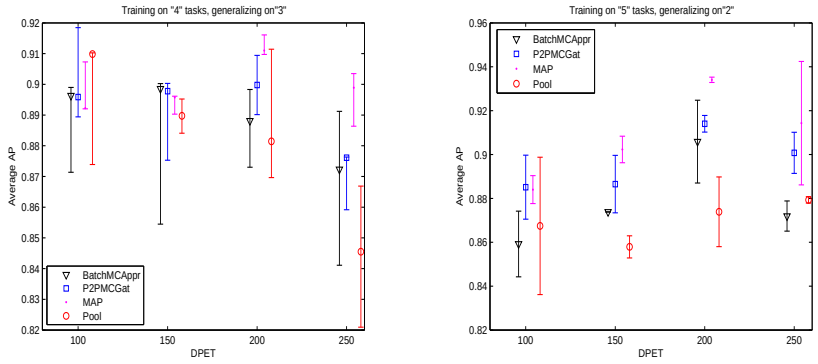
Figure 12: Average AP on the 11 unseen tasks of Landmine data set; training on 8 tasks, generalising on 11; (a) Overall AP over 11 tasks, (b) Average AP over 6 tasks of the first cluster, (c) Average AP over 5 tasks of the second cluster.

4.2.3 ARRHYTHMIA CLASSIFICATION

As a second real data set, we return to the arrhythmia classification problem introduced in Section 4.1.4. The results from the fully observed tasks scenario indicate an unclear pattern of correlations between the tasks, as summarised in the task covariance matrix Figure 7.b, which calls into question the validity of the similarity of distribution assumption. Fortunately, in this application the classes have a physical interpretation. For example normal heart beats between different patients, although not exactly the same, can be expected to be similar, and a PVC arrhythmic heart beat of one patient can not have the wave form of a normal heart beat from another patient. This allows us to assume that the classes between the tasks will not be anti-correlated, so that at least the low-error joint prediction assumption should approximately hold.

Since there are no obvious clusters among tasks, in this set of experiments the training tasks are chosen by randomly selecting some for training and keeping the rest as test tasks. Figure 13 presents the results on the unseen tasks that were obtained by training the CMCMT model with 4 and 5 tasks. First of all, we observe that the average AUC in the partially observed case is a

lot lower than in the fully observed case, something perhaps to be expected since, contrary to the previous two examples, the model assumptions are not fully met in this data set. Surprisingly, the method that achieved the best performance was the MAP, and no principled justification can be given for that. Secondly, we observe that the performance in this set of experiments exhibits some interesting patterns as the number of training tasks increases. Specifically, for four training tasks the performance of all methods does not significantly improve as we increase the number of data points per task, and in some cases it even deteriorates, a phenomenon that was also observed for 2 and 3 training tasks but results are omitted for brevity. This indicates that if the space of tasks has not been sampled sufficiently, the model can not yield good generalisation performance to new tasks, even if the number of training data increases. In contrast, for five training tasks the MAP and P2PGat methods yield a significant improvement of performance as the number of data points increases (levelling off after 200 DPETs).



(a) (b)

Figure 13: Average AP on the unseen tasks of Arrhythmia data set on different number of training tasks; (a) training on 4 tasks, generalising on 3, (b) training on 5 tasks, generalising on 2.

Empirically, it would appear that the P2PGat method is preferable to the Batch method when the model assumptions are violated. Intuitively, one could argue that the Batch method is less flexible, as the relative contribution of the different single-class predictors is fixed across all points in the target task. Therefore, if the model assumptions are violated, leading to an incorrect task labelling, the propagated error could have a worse effect in Batch than in P2PGat. This is partly confirmed by the analysis of Toy data set *II* in Section 4.1.2, where the model assumptions were violated and P2PGat gave significantly higher AP than the Batch method.

4.2.4 CHARACTER CLASSIFICATION

For reasons of completeness, we present an analysis of the character classification problem in the partially observed tasks scenario. Here the validity of the model assumptions is dubious; nevertheless, we believe that interesting lessons can be learned from model failure. The fully observed tasks analysis of the character classification problem did not reveal any clusters of tasks. Furthermore, there is no reason to believe that the low-error joint prediction assumption may hold: some tasks might even be anticorrelated, as in tasks ‘a/g’ and ‘g/y’, where letter ‘g’ belongs to the negative class

for task ‘a/g’, and to the positive class for task ‘g/y’. Therefore, the character classification problem is ill-suited for this type of experiments. This is borne out by experimental evidence: simulation results with 4, 5, and 6 training tasks, which are omitted for brevity, indicated that increasing the number of tasks and the number of training points per task does not improve the performance in any of the methods. Specifically, the results obtained were close to that of a random predictor indicating statistical inconsistency of the model assumptions with the data.

4.2.5 OBSERVATIONS

Meta-generalising in a partially observed tasks scenario is an extremely hard problem; nevertheless, we believe there are some interesting points that can be made from the previous experimental analysis. Below we summarise the most important observations for this scenario.

1. In situations where there are clusters of tasks, even though the model hasn’t seen all tasks, the Batch method can still make accurate predictions that reaches the performance of the fully observed tasks case. Pragmatically, one could consider whether the training phase of the model has revealed clusters of tasks when deciding which prediction method to apply.

2. In multi-task problems where the correlations between the tasks are less pronounced, but where the low-error joint prediction is satisfied and where a sufficient number of training tasks is available, the method that is most appropriate is the P2PGat, since it provides a more flexible task assignment mechanism than the Batch mode. The validity of the low-error joint prediction assumption can sometimes be assessed from the nature of the problem (as in the arrhythmia case).

3. Sufficient exploration of the task space is essential for the success of the method. While we have not tested our model for very large numbers of training tasks, the results suggest that often a significant improvement in performance can be achieved when the number of training tasks crosses a critical number, indicating a sufficient coverage of the task space. This phenomenon was observed in the Arrhythmia classification problem for 2 and 3 training tasks where the performance of the models remained the same as the number of training samples per task increased. In essence more training data lead to stronger biases for meta-generalisation in target tasks that are not correlated with any of the training tasks.

4. In most cases, when the assumptions of the model are only approximately met and when the exploration of the task space is insufficient, the generalisation performance on totally unseen tasks is still modest, and it may be that other approaches based on mixtures of GP experts (Tresp, 2000) achieve similar results. An extensive comparison with these approaches would be interesting, but outside the scope of the present work.

5. Conclusions

In this paper we presented an investigation on the use of Gaussian Processes for meta-generalisation, that is, predicting on unseen learning tasks by leveraging the information of several, related tasks. Our model attacks the meta-generalisation problem by coupling two GPs, a multi-class classifier that learns task responsibilities, and a multi-task classifier that learns prediction models on individual tasks as well as learning the global correlation structure between training tasks. While it

should be emphasized that this is an initial attempt to address what is certainly a very ambitious problem, we believe the model will prove useful to understand meta-generalisation. First of all, it provides a constructive approach to meta-generalisation: most previous studies (Baxter, 2000) have been mainly theoretical investigations attempting to establish the necessary conditions for meta-generalisation to work, or have focused on the domain adaptation scenario (Ben-David et al., 2007, 2010). Our model is an attempt to translate these conditions into a model, and to investigate how well such a model may perform on real meta-generalisation problems.

It is important to remark that our method crucially relies on the ability to learn the covariance matrix of a GP: the fundamental ingredient in this work is the task correlation matrix which captures the correlations between source tasks. This not only has a significant impact on the prediction results, but can reveal the presence of clusters of tasks within the data, hence guiding the choice of the appropriate prediction method (Batch or P2PGat). Many multi-task learning approaches do not explicitly model the correlations, but transfer learning solely through some shared prior over parameters (Yu et al., 2005, e.g.). While this could have computational advantages, we would argue that the implicit modelling of task correlations would make them less suitable for meta-generalisation. A common problem, shared by many GP models, is the computational cost when samples become large, which would be the probable situation in many applications such as personalised medicine. Our approach also suffers from the cubic scaling of matrix inversions needed within GP inference; while sparsity inducing approaches could be helpful (Snelson and Ghahramani, 2006), it would be interesting to explore sparsity within the task space as well as within the data space.

While we believe that our results are encouraging and help clarify the importance of the various assumptions underlying meta-generalisation, it remains undeniable that in many practical situations it is impossible to assess the validity of these assumptions, making meta-generalisation an extremely challenging problem. Possible avenues to extend the applicability of the approach could be to consider task descriptor features, or to introduce a semi-supervised element in the model in the spirit of domain adaptation approaches.

Acknowledgments

We would like to acknowledge support for this project from the Engineering and Physical Sciences Research Council (EPSRC) under grant EP/F009461/2. G. Sanguinetti acknowledges support from the Scottish government through the Scottish Informatics and Computer Science Alliance (SICSA). We wish to thank Eleni Vasilaki, Amos Storkey, Sethu Vijayakumar and Mark Girolami for the useful discussions and their support.

Appendix A. Approximate Inference

This appendix computes the approximate posteriors for $Q(\mathbf{g})$, $Q(\mathbf{f})$ and $Q(\mathbf{h}^x)$. The posterior of $Q(\mathbf{H}^t)$ can be found in Girolami and Rogers (2006) and therefore details are omitted.

A.1 $Q(\mathbf{g})$

The approximate posterior for $Q(\mathbf{g})$ is computed as Girolami and Rogers (2006)

$$\begin{aligned} Q(\mathbf{g}) &\propto \exp\left\{\mathbb{E}_{Q(\mathbf{h}^t)}\left(\sum_{i=1}^{N}\sum_{j=1}^{M}\log p(h_{ij}^t|g_{ij}) + \log p(\mathbf{g}_j|\mathbf{X})\right)\right\} \\ &\propto \exp\left\{\mathbb{E}_{Q(\mathbf{h}^t)}\left(\sum_{j=1}^{M}\log \mathcal{N}_{\mathbf{h}_j^t}(\mathbf{g}_j,\mathbf{I}) + \log \mathcal{N}_{\mathbf{g}_j}(0,\mathbf{K}^x)\right)\right\} \\ &\propto \prod_{j=1}^{M}\mathcal{N}_{\mathbf{h}_j^t}(\mathbf{g}_j,\mathbf{I})\mathcal{N}_{\mathbf{g}_j}(0,\mathbf{K}^x), \end{aligned}$$

which gives that $Q(\mathbf{g}) = \prod_{j=1}^{M} Q(\mathbf{g}_j) = \prod_{j=1}^{M}\mathcal{N}_{\mathbf{g}_j}(\tilde{\mathbf{g}}_j,\mathbf{\Sigma}^g)$, where $\mathbf{\Sigma}^g = \left(\mathbf{I}+(\mathbf{K}^x)^{-1}\right)^{-1} = \mathbf{K}^x(\mathbf{I}+\mathbf{K}^x)^{-1}$, and $\tilde{\mathbf{g}}_j = \mathbf{\Sigma}^g\tilde{\mathbf{h}}_j^t$.

A.2 $Q(\mathbf{f})$

$$\begin{aligned} Q(\mathbf{f}) &\propto \exp\left\{\mathbb{E}_{Q(\mathbf{h}^x)}\left\{\sum_{i=1}^{NM}\log p(h_i^x|f_i) + \log p(\mathbf{f}|\mathbf{X})\right\}\right\} \\ &\propto \exp\left\{\mathbb{E}_{Q(\mathbf{h}^x)}\left\{-\frac{1}{2}\mathbf{h}^{xT}\mathbf{h}^x + \mathbf{f}^T\mathbf{h}^x - \frac{1}{2}\mathbf{f}^T\mathbf{f} - \frac{1}{2}\mathbf{f}^T\left(\mathbf{K}^t\otimes\mathbf{K}^x\right)^{-1}\mathbf{f} + const.\right\}\right\} \\ &\propto \exp\left\{-\frac{1}{2}\mathbf{f}^T(\mathbf{I}+\left(\mathbf{K}^t\otimes\mathbf{K}^x\right)^{-1})\mathbf{f} + \mathbf{f}^T\tilde{\mathbf{h}} + const.\right\}, \end{aligned}$$

which gives that $Q(\mathbf{f}) = \mathcal{N}_{\mathbf{f}}(\tilde{\mathbf{f}},\mathbf{\Sigma}^f)$ where $\tilde{\mathbf{f}} = \mathbf{\Sigma}^f\tilde{\mathbf{h}}^x$, and $\mathbf{\Sigma}^f = (\mathbf{I}+(\mathbf{K}^t\otimes\mathbf{K}^x)^{-1})^{-1} = \mathbf{K}^t\otimes\mathbf{K}^x(\mathbf{I}+\mathbf{K}^t\otimes\mathbf{K}^x)^{-1}$.

A.3 $Q(\mathbf{h}^x)$

$$\begin{aligned} Q(\mathbf{h}^x) &\propto \exp\left\{\mathbb{E}_{Q(\mathbf{f})}\left\{\sum_{i=1}^{NM}\log p(y_i^x|h_i^x) + \log p(h_i^x|f_i)\right\}\right\} \\ &\propto \exp\left\{\log\left(\prod_{i=1}^{NM}p(y_i^x|h_i^x)\right) + \log\left(\prod_{i=1}^{NM}\mathcal{N}_{h_i^x}(\tilde{f}_i,1)\right)\right\} \\ &\propto \prod_{i=1}^{NM}\mathcal{N}_{h_i^x}(\tilde{f}_i,1)\delta(h_i^x) \end{aligned}$$

which gives that $Q(h_i^x) = \frac{1}{Z_i}\mathcal{N}_{h_i^x}(\tilde{f}_i,1)\delta(h_i^x)$, and we have that

$$Q(h_i^x) = \left\{\begin{array}{l} \frac{1}{Z_i}\int_0^{+\infty} h_i^x\mathcal{N}_{h_i^x}(\tilde{f}_i,1) = \tilde{f}_i + \frac{\mathcal{N}_{\tilde{f}_i}(0,1)}{\Phi(\tilde{f}_i)} \;\text{ for }\; y_i^x = +1 \\ \\ \frac{1}{Z_i}\int_{-\infty}^{0} h_i^x\mathcal{N}_{h_i^x}(\tilde{f}_i,1) = \tilde{f}_i - \frac{\mathcal{N}_{\tilde{f}_i}(0,1)}{\Phi(-\tilde{f}_i)}) \;\text{ for }\; y_i^x = -1 \end{array}\right\},$$

where $Z_i = \Phi(\pm\tilde{f}_i)$ for $y_i^x = \pm 1$. The approximate posterior of $Q(\mathbf{H}^t)$ can be computed in a similar manner, and we refer the interested reader to Girolami and Rogers (2006).

Appendix B. Lower Bound

This appendix presents the analytical form of the variational bound as well as the gradients of the bound with respect to the hyperparameters θ^x and θ^t.

B.1 Lower Bound on Log Marginal Likelihodd

The lower bound on the log marginal likelihood is computed by

$$\begin{aligned}\mathcal{L}(Q) &= \mathbb{E}_{Q(\Theta)}[\log p(\mathbf{y}^t,\mathbf{y}^x,\mathbf{g},\mathbf{h}^t,\mathbf{f},\mathbf{h}^x|X,\theta^t,\theta^x)] - \\ &\qquad \mathbb{E}_{Q(\Theta)}[\log Q(\mathbf{g})Q(\mathbf{h}^t)Q(\mathbf{f})Q(\mathbf{h}^x)] \\ &= -\frac{NM}{2}\log(2\pi) + \frac{N}{2}\log(2\pi) + \frac{NM}{2} - \frac{M}{2}\text{trace}(\mathbf{\Sigma}^g) - \frac{1}{2}\sum_m \tilde{\mathbf{g}}_m^T \mathbf{K}^{x^{-1}}\tilde{\mathbf{g}}_m \\ &- \frac{M}{2}\text{trace}\left(\mathbf{K}^{x^{-1}}\mathbf{\Sigma}^g\right) - \frac{M}{2}\log|\mathbf{K}^x| + \frac{M}{2}\log|\mathbf{\Sigma}^g| + \sum_n \log z_n^t \\ &+ \sum_{n=1}^N \log z_n^x - \frac{1}{2}\log|\mathbf{I}+\mathbf{K}^t\otimes\mathbf{K}^x| - \frac{1}{2}\tilde{\mathbf{f}}^T(\mathbf{K}^t\otimes\mathbf{K}^x)^{-1}\tilde{\mathbf{f}}, \end{aligned} \tag{15}$$

where $z_n^t = \mathbb{E}_{p(u)}\left\{\prod_{j\neq i}\Phi(u+\tilde{g}_{ni}-\tilde{g}_{nj})\right\}$, and $z_n^x = \Phi(y_n^x \tilde{f}_n)$.

Terms that depend on hyperparameters θ^x and θ^t are:

$$\begin{aligned}\mathcal{L}(Q)_{\theta^x,\theta^t} &= -\frac{M}{2}\text{trace}(\mathbf{\Sigma}^g) - \frac{1}{2}\sum_m \tilde{\mathbf{g}}_m^T \mathbf{K}^{x^{-1}}\tilde{\mathbf{g}}_m - \frac{M}{2}\text{trace}\left(\mathbf{K}^{x^{-1}}\mathbf{\Sigma}^g\right) \\ &- \frac{M}{2}\log|\mathbf{K}^x| + \frac{M}{2}\log|\mathbf{\Sigma}^g| - \frac{1}{2}\log|\mathbf{I}+\mathbf{K}^t\otimes\mathbf{K}^x| - \frac{1}{2}\tilde{\mathbf{f}}^T(\mathbf{K}^t\otimes\mathbf{K}^x)^{-1}\tilde{\mathbf{f}}.\end{aligned}$$

B.2 Gradients on Lower Bound

The gradients with respect to the parameters of the data covariance function $\mathbf{K}^x$ are computed from:

$$\begin{aligned}\frac{\partial}{\partial\theta^x}\mathcal{L}(q) &= -\frac{M}{2}\text{trace}\left\{\mathbf{\Omega}(\mathbf{I}+\mathbf{K}^x)^{-1} - \mathbf{K}^x(\mathbf{I}+\mathbf{K}^x)^{-1}\mathbf{\Omega}(\mathbf{I}+\mathbf{K}^x)^{-1}\right\} + \frac{1}{2}\tilde{\mathbf{g}}_m^T\mathbf{K}^{x^{-1}}\mathbf{\Omega}\mathbf{K}^{x^{-1}}\tilde{\mathbf{g}}_m \\ &+ \frac{M}{2}\text{trace}\left\{(\mathbf{I}+\mathbf{K}^x)^{-1}\mathbf{\Omega}(\mathbf{I}+\mathbf{K}^x)^{-1}\right\} - \frac{M}{2}\text{trace}\left\{\mathbf{K}^{x^{-1}}\mathbf{\Omega}\right\} \\ &+ \frac{M}{2}\text{trace}\left\{(\mathbf{I}+\mathbf{K}^{x^{-1}})^{-1}\mathbf{K}^{x^{-1}}\mathbf{\Omega}\mathbf{K}^{x^{-1}}\right\} + \frac{1}{2}\tilde{\mathbf{f}}^T\left(\mathbf{K}^t\otimes\mathbf{K}^x\right)^{-1}\mathbf{K}^t\otimes\mathbf{\Omega}\left(\mathbf{K}^t\otimes\mathbf{K}^x\right)^{-1}\tilde{\mathbf{f}} \\ &- \frac{1}{2}\text{trace}\left(\left(\mathbf{I}+\mathbf{K}^t\otimes\mathbf{K}^x\right)^{-1}\mathbf{K}^t\otimes\mathbf{\Omega}\right).\end{aligned} \tag{16}$$

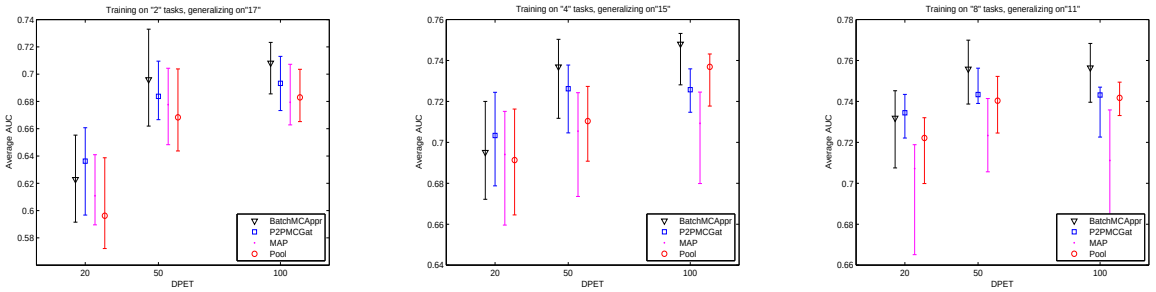
While the gradients with respect to the parameters of the task covariance matrix are computed from:

$$\frac{\partial}{\partial\theta^t}\mathcal{L}(q) = \frac{1}{2}\tilde{\mathbf{f}}^T\left(\mathbf{K}^t\otimes\mathbf{K}^x\right)^{-1}\mathbf{\Xi}\otimes\mathbf{K}^x\left(\mathbf{K}^t\otimes\mathbf{K}^x\right)^{-1}\tilde{\mathbf{f}} - \frac{1}{2}\text{trace}\left(\left(\mathbf{I}+\mathbf{K}^t\otimes\mathbf{K}^x\right)^{-1}\mathbf{\Xi}\otimes\mathbf{K}^x\right),$$

where $\mathbf{\Omega} = \frac{\partial\mathbf{K}^x}{\partial\theta^x}$, and $\mathbf{\Xi} = \frac{\partial\mathbf{K}^t}{\partial\theta^t}$

Appendix C. Additional Results on the Landmine Detection Problem

This appendix provides additional results for the Landmine detection problem (section 4.2.2) from the Partially observed tasks scenario. In contrast to the results presented in section 4.2.2 where methods were compared in terms of AP, Figure 14 presents results in terms of AUC, similarly to previous studies in that data set (Xue et al., 2007; Liu et al., 2009).



(a) (b) (c)

Figure 14: AUC on the Landmine detection problem; (a) AUC over 17 tasks by training on 2 tasks, (b) AUC over 15 tasks by training on 4 tasks, (c) AUC over 11 tasks by training on 8 tasks.

References

J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993.

R.K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research*, 6:1817–1853, 2005.

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

A. Arnold, R. Nallapati, and W.W. Cohen. A comparative study of methods for transductive transfer learning. In *Proceedings of the 7th IEEE International Conference on Data Mining Workshops*, pages 77–82, Omaha, Nebraska, USA, 2007.

B. Bakker and T. Heskes. Task clustering and gating for bayesian multitask learning. *The Journal of Machine Learning Research*, 4:83–99, 2003.

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

S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems 19*, pages 137–145, Vancouver, Canada, 2007.

S. Ben-David, T. Luu, T. Lu, and D. Pál. Impossibility theorems for domain adaptation. In *Proceedings of the 13th International Workshop on Artificial Intelligence and Statistics*, volume 13, pages 129–136, Sardinia, Italy, 2010.

S. Bickel, M. Brückner, and T. Scheffer. Discriminative learning under covariate shift. *The Journal of Machine Learning Research*, 10:2137–2155, 2009.

E. Bonilla, K. M. Chai, and C.K.I. Williams. Multi-task gaussian process prediction. In *Advances in Neural Information Processing Systems 20*, pages 153–160, Vancouver, Canada, 2008.

K.H. Brodersen, C.S. Ong, K.E. Stephan, and J.M. Buhmann. The binormal assumption on precision-recall curves. In *Proceedings of the 2010 International Conference on Pattern Recognition*, pages 4263–4266, Istanbul, Turkey, 2010.

R. Caruana. Multi-task learning. *Machine Learning*, 28(1):41–75, 1997.

O. Chapelle, B. Schölkopf, and A. Zien. *Semi-supervised learning*. MIT Press, Cambridge, MA, 2006.

K. Crammer, M. Kearns, and J. Wortman. Learning from multiple sources. *The Journal of Machine Learning Research*, 9:1757–1774, 2008.

N. A.C. Cressie. *Statistics for Spatial Data*. John Wiley & Sons. New York. US, 1993.

L. Csató, E. Fokoué, M. Opper, B. Schottky, and O. Winther. Efficient approaches to gaussian process classification. In *Advances in Neural Information Processing Systems 12*, pages 251–257, Denver, Colorado, 2000.

H. Daumé. Frustratingly easy domain adaptation. In *Annual Meeting of the Association for Computational Linguistics*, volume 45, pages 256–263, 2007.

H. Daumé III. Bayesian multitask learning with latent hierarchies. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pages 135–142, Montreal, Canada, 2009.

H. Daumé III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26(1):101–126, 2006.

J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 233–240, Pittsburgh, USA, 2006.

I. Deak. Three digit accurate multiple normal probabilities. *Numerische Mathematik*, 35(4):369–380, 1980.

H. I. Gassmann, I. Deak, and T. Szantai. Computing multivariate normal probabilities: A new look. *Journal of Computational and Graphical Statistics*, 11(4):920–949, 2002.

A. Genz. Numerical computation of multivariate normal probabilities. *Journal of Computational and Graphical Statistics*, 1(2):141–149, 1992.

M. Girolami and S. Rogers. Variational bayesian multinomial probit regression with gaussian process priors. *Neural Computation*, 18(8):1790–1817, 2006.

M. Girolami and M. Zhong. Data integration for classification problems employing Gaussian process priors. In *Advances in Neural Information Processing Systems 19*, pages 465–472, Vancouver, Canada, 2007.

A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23): 215–220", 2000.

A. K. Gupta and D. K. Nagar. *Matrix Variate Distributions*. Chapman & Hall/CRC, 2000.

J. A. Hanley and B. J. Mcneil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36, April 1982.

G.E. Hinton. Connectionist learning procedures. *Artificial Intelligence*, 40(1-3):185–234, 1989.

J. Huang, A. J. Smola, A. Gretton, K M. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems 19*, pages 601–608, Vancouver, Canada, 2007.

R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and G.E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

L.I. Kuncheva. A theoretical study on six classifier fusion strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):281–286, 2002.

Q. Liu, X. Liao, H. Li, J. R. Stack, and L. Carin. Semisupervised multitask learning. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 31(6):1074–1086, 2009.

D.J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems 21*, pages 1041–1048, Vancouver, Canada, 2009.

T.P. Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, volume 17, pages 362–369, San Francisco, CA, USA, 2001.

M. Opper and O. Winther. Gaussian processes for classification: mean-field algorithms. *Neural Computation*, 12(11):2655–2684, 2000.

S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

S.J. Pan, I.W. Tsang, J.T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2009.

R. Raina, A. Battle, H. Lee, B. Packer, and A.Y. Ng. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the 24th International Conference on Machine Learning*, pages 759–766, Corvallis, OR, USA, 2007.

C. E. Rasmussen and C. K.I. Williams. *Gaussian Processes for Machine Learning*. MIT press, 2005.

C.E. Rasmussen and Z. Ghahramani. Infinite mixtures of gaussian process experts. In *Advances in Neural Information Processing Systems 14*, pages 881–888, Vancouver, Canada, 2001.

R. Rebonato and P. Jäckel. The most general methodology to create a valid correlation matrix for risk management and option pricing purposes. *Journal of Risk*, 2(2), 2000.

G. Skolidis and G. Sanguinetti. Bayesian multitask classification with gaussian process priors. *IEEE Transactions on Neural Networks*, 22(12):2011 –2021, Dec. 2011.

G. Skolidis, RH Clayton, and G. Sanguinetti. Automatic classification of arrhythmic beats using gaussian processes. In *IEEE Transactions on Computers in Cardiology, 2008*, pages 921–924, Bologna, Italy, 2008.

E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18*, pages 1257–1264, Vancouver, Canada, 2006.

A. J. Storkey and M. Sugiyama. Mixture regression for covariate shift. In *Advances in Neural Information Processing Systems 19*, pages 1337–1344, Vancouver, Canada, 2007.

M. Sugiyama, M. Krauledat, and K.R. Müller. Covariate shift adaptation by importance weighted cross validation. *The Journal of Machine Learning Research*, 8:985–1005, 2007.

Volker Tresp. Mixtures of gaussian processes. In *Advances in Neural Information Processing Systems 13*, pages 654–660, Vancouver, Canada, 2000. MIT Press.

S.R. Waterhouse. *Classification and Regression Using Mixtures of Experts*. PhD thesis, Department of Engineering, Cambridge University, 1997.

Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *The Journal of Machine Learning Research*, 8:35–63, 2007.

K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 1012–1019, Bonn, Germany, 2005.