22

Life-long Learning Through Task Rehearsal and Selective Knowledge Transfer

Daniel L. Silver[1] and Robert E. Mercer[2]
[1]*Jodrey School of Computer Science, Acadia University*
[2] *Department of Computer Science, The University of Western Ontario*
Canada

1. Introduction

The majority of machine learning research has focused on the single task learning (STL) approach where an hypothesis for a single task is induced from a set of supervised training examples. In contrast, one of the key aspects of human learning is that individuals face a sequence of learning problems over a lifetime. Humans take advantage of this by transferring knowledge from previously learned tasks to facilitate the learning of a new task. *Life-long learning*, a relatively new area of machine learning research, is concerned with the persistent and cumulative nature of learning (Thrun, 1997). Life-long learning considers situations in which a learner faces a series of different tasks and develops methods of retaining and using prior knowledge to improve the effectiveness (more accurate hypotheses) and efficiency (shorter training times) of learning. Related names for life-long learning in the literature are *learning to learn* and *meta-learning*.

A challenge often faced by a life-long learning agent is a deficiency of training examples from which to develop accurate hypotheses. Machine learning theory tells us that this problem can be overcome with an appropriate *inductive bias* (Mitchell, 1997), one source being *prior task knowledge* (Baxter, 1995). Lacking a theory of knowledge transfer (Caruana, 1997, Thrun, 1997) that distinguishes knowledge from related and unrelated tasks, we have developed one and applied it to life-long learning problems, such as learning a more accurate medical diagnostic model from a small sample of patient data (Silver, 2000). The approach requires (1) a method of selectively transferring previously learned knowledge to a new task based on a measure of task relatedness and (2) a method of retaining learned task knowledge and its recall when learning a new task.

In (Silver & Mercer, 1996) we introduced ηMTL, a modified version of the multiple task learning (MTL) method of functional transfer to provide a solution to the first problem of selective transfer. Using a measure of previously learned secondary task to primary task relatedness, an ηMTL network can favourably bias the induction of a hypothesis for a primary task. Section 3 reviews the necessary aspects of ηMTL.

This paper focuses on the Task Rehearsal Method (TRM) to solve the second problem of retention and recall of learned task knowledge. TRM uses either the standard MTL or the ηMTL learning algorithms as the method of knowledge transfer and inductive bias. Task rehearsal is so named because previously learned tasks are relearned or *rehearsed* in parallel

Source: Theory and Novel Applications of Machine Learning, Book edited by: Meng Joo Er and Yi Zhou,
ISBN 978-3-902613-55-4, pp. 376, February 2009, I-Tech, Vienna, Austria

with the learning of each new task. It is through the rehearsal of previously learned tasks that the inductive bias of prior knowledge influences the hypothesis for the new task.
The contributions of this work is the demonstration that task knowledge can be effectively and efficiently retained within a neural network representation, that this knowledge can be selectively and accurately transferred using the TRM and ηMTL approach, and that a life-long sequential learning system can be developed based on the approach.
The following section provides appropriate background on knowledge based inductive learning, inductive bias, knowledge transfer with MTL neural networks and the origins of task rehearsal. Section 3 reviews selective knowledge transfer with ηMTL using a measure of task relatedness. Section 4 develops the TRM of life-long learning and discusses a prototype software system. Section 5 discusses the results of empirical studies using TRM and ηMTL on four domains of tasks. Section 6 presents the important findings made while implementing and testing the prototype system, reviews closely related work by other researchers and suggests future work in this area. Finally, Section 7 concludes with a summary of the paper.

2. Background

2.1 Knowledge based inductive learning

The constraint on a learning system's hypothesis space, beyond the criterion of consistency with the training examples, is called *inductive bias* (Mitchell, 1980). An inductive bias of a learning system can be expressed as the system's preference for one hypothesis over another, for example Occam's Razor suggests a bias for simple over more complex hypotheses. Inductive bias is essential for the development of a hypothesis with good generalization from a practical number of examples (Mitchell, 1997). Ideally, a life-long learning system can select its inductive bias to tailor the preference for hypotheses according to the task being learned (Utgoff, 1986). One type of inductive bias is knowledge of the task domain. The retention and use of domain knowledge as a source of inductive bias remains an unsolved problem in machine learning.
We define *knowledge based inductive learning* as a learning method which relies on prior knowledge of the problem domain to reduce the hypothesis space which must be searched. Figure 1 provides the framework for knowledge based inductive learning. *Domain knowledge* is a database of accumulated information which has been acquired from previously learned tasks. The intent is that this knowledge will bias a pure inductive learning system in a positive manner such that it trains in a shorter period of time and produces a more accurate hypothesis with a fewer number of training examples. In turn, new information is added to, or *consolidated* within the domain knowledge database following its discovery. Michalski refers to this as *constructive inductive learning* (Michalski, 1993). In the extreme, where the new classification task to be learned is exactly the same as one learned at some earlier time, the inductive bias should provide rapid convergence on the optimal hypothesis with very few examples. Formally, given a learning algorithm L and a domain knowledge inductive bias B_D, the problem becomes one of finding a hypothesis h, based on a set of examples $S = (x_i, t_i)$ from a concept space X, such that:

$$L \wedge B_D \wedge S \succ h$$

where $h(x_i) = t_i$ for all (x_i, t_i) in X and $\succ$ means inductive inference. The relation is not one of deductive inference because it is possible that B_D forms only a portion of all assumptions required to logically deduce h given S.

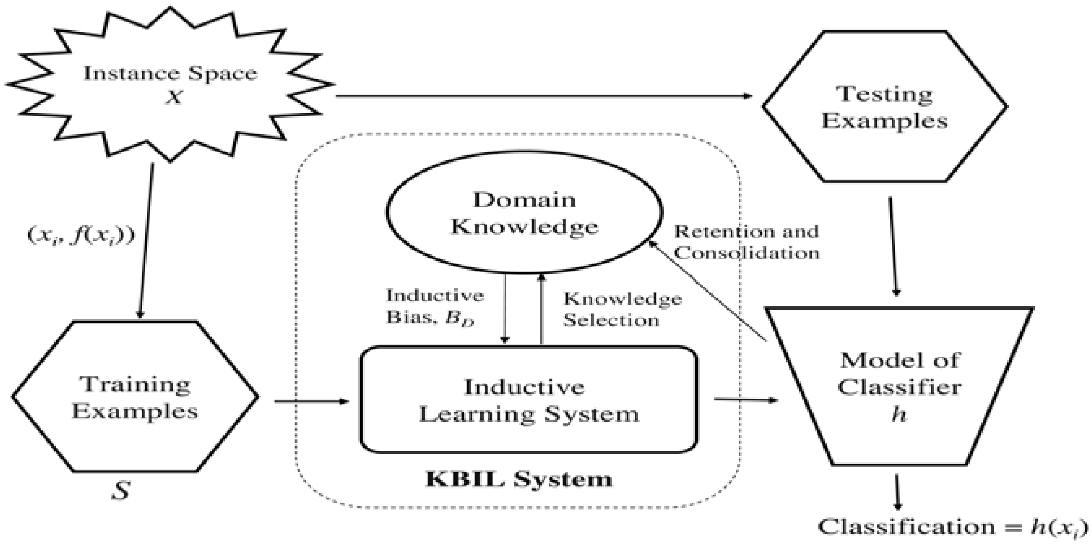


Fig. 1. The framework for knowledge based inductive learning.

When task domain knowledge is used to bias an inductive learner, a *transfer* of knowledge occurs from one or more *source* or *secondary* tasks to a *target* or *primary* task. Thus, the problem of selecting an appropriate bias is transformed into the problem of selecting the appropriate task knowledge for transfer.

The problem of knowledge transfer is an important aspect of life-long learning. For a good survey of knowledge transfer methods see (Pratt & Jennings, 1996), for a related survey on learning to learn read (Thrun & Pratt, 1997), and for a recent survey on metalearning see (Vilatlta & Drissi, 2002). Each of these surveys conclude that a significant problem in using prior knowledge is the selection of appropriate related knowledge when learning a new task.

2.2 Representational vs. functional transfer

In (Silver & Mercer, 1996) we define the difference between two forms of task knowledge transfer: *representational* and *functional*. The *representational* form of transfer involves the direct or indirect assignment of known task representation (weight values) to a new task. We consider this to be an explicit form of knowledge transfer from a source task to a target task. Since 1990 numerous authors have discussed methods of representational transfer (Fahlman & Lebiere, 1990, Pratt, 1993, Ring, 1993, Sharkey & Sharkey, 1992, Shavlik & Towell, 1990, Singh, 1992, Towell et al., 1990) which often results in substantially reduced training time with no loss in generalization performance.

In contrast to representational transfer is a form we define as *functional*. Functional transfer does not involve the explicit assignment of prior task representation to a new task, rather it employs the use of implicit *pressures* from supplemental training examples (Abu-Mostafa, 1995, Suddarth & Kergoisien, 1990), the parallel learning of related tasks constrained to use a common internal representation (Baxter, 1995, Caruana, 1997), or the use of historical training information (most commonly the learning rate or gradient of the error surface) to augment the standard weight update equations (Mitchell & Thrun, 1993, Naik & Mammone, 1993, Thrun, 1995). These pressures serve to reduce the effective hypothesis space in which the learning system performs its search. This form of transfer has its greatest value from the perspective of increased generalization performance. Certain methods of functional transfer have also been found to reduce training time (measured in number of training iterations). Chief among these methods is the parallel multiple task learning (MTL) paradigm which is discussed in the next subsection.

The form of knowledge transfer can be independent of the form of knowledge retention. For example, functional knowledge of a task can be retained in the form of training examples and later used to transfer knowledge when learning a new task. Alternatively, the representation of a task model can be retained and later used to generate training examples that can then be used for functional transfer.

2.3 Multiple Task Learning (MTL)

Psychological studies of human and animal learning conclude that besides the development of a specific discriminant function which satisfies the task at hand, there is the acquisition of general knowledge of the task domain. This general knowledge remains available for use in subsequent learning (Kehoe, 1988). This concept has been formalized by Baxter's work on learning internal representations (Baxter, 1995) and demonstrated by Caruana (Caruana, 1997) through a method called multiple task learning (MTL). We classify MTL as a functional form of knowledge transfer.

An MTL network uses a feed-forward multi-layer network with an output for each task to be learned. Figure 2 shows a simple MTL network containing a hidden layer of nodes, henceforth referred to as the *common feature layer*, that are shared by all tasks. The sharing of the *internal representation* (the weights of connections) below the common feature layer is the method by which inductive bias occurs within an MTL network. This is a powerful method of knowledge transfer. For example, a two output MTL network can learn the logical XOR and ¬XOR functions because they share a common internal representation. By comparison, it is not possible to learn these two tasks within the same single task learning (STL) network because their examples would conflict.

MTL training examples are composed of a set of input attributes as well as a target output for each task. The standard back-propagation of error learning algorithm is used to train all

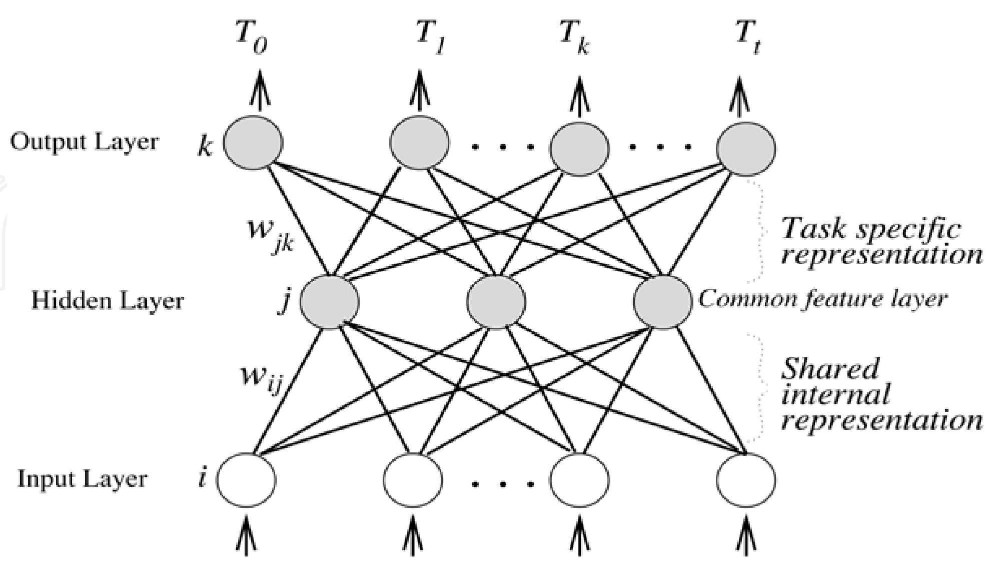


Fig. 2. A multiple task learning (MTL) network. There is an output node for each task being learned in parallel. The representation formed in the lower portion of the network is common to all tasks.

tasks in parallel. The weights, w_{jk}, affecting an output node k are adjusted according to the equation: $\Delta w_{jk} = \eta \frac{\partial E_k}{\partial w_{jk}}$; where η is the learning rate parameter, E_k is the error (or cost) function being minimized and $\frac{\partial E_k}{\partial w_{jk}}$ is the error signal that is propagated backward through the network. Under the standard back-propagation algorithm η is a constant, global parameter for all tasks. Consequently, the back-propagated error signal from any output node k is considered to be of equal value to all others. At the lowest training error, the back-propagation algorithm generates hypotheses that average the error across all of the output nodes. The averaging effect may not be beneficial for every task (Caruana, 1997).
Baxter has proven that the number of examples (sample complexity) required for learning any one task using an MTL network decreases as a function of the total number of related tasks being learned in parallel (Baxter, 1995). Baxter has also proven that the common internal representation acquired will facilitate the learning of subsequent related tasks sampled from the domain. In an MTL network this translates into a common representational component developed within the input to hidden weights for all tasks (see Figure 2). For any particular task the hidden to output weights constitute the task specific component. Since the number of weights in this section of the network is relatively small, the training of a new task from the domain can be accomplished with relatively few examples and with a smaller amount of effort compared with single task learning.
Functional transfer and positive inductive bias occurs in an MTL network due to the pressures of learning several related tasks under the constraint that the majority of the connection weights of each task are shared. Therefore, to ensure the functional transfer of knowledge from several secondary tasks to the primary task: (1) the MTL network should have a sufficient amount of internal representation (the sum of the hidden nodes required for learning each task under STL is suggested by (Caruana, 1997)), and (2) the secondary tasks should be as closely related to the primary task as possible.

2.4 Rehearsal of task examples

A fundamental problem with using back-propagation neural networks as the basis for a life-long learning system is the phenomenon of *catastrophic interference* (Grossberg, 1987, McCloskey & Cohen, 1989). Consider the problem of learning one set of randomly chosen input-output examples by an STL ANN and subsequently learning another set of examples using the same ANN. Catastrophic interference occurs as the hypothesis for the second set of examples interferes with the existing hidden node representation developed for the first set. The result is that knowledge of the first set of examples is “forgotten”. Psychologists have long considered this a major failing of ANN models of long-term memory.
Rehearsal and *pseudo-rehearsal* of examples is presented as solutions to catastrophic interference in (Robins, 1995, 1996). Robins shows that the existing hidden node representation within an STL network can be maintained by relearning or rehearsing a subset of the previously learned examples while concurrently learning the new set of examples. Given sufficient internal representation (hidden nodes), an appropriate model will develop such that it satisfies the requirements of both set of examples to the extent to which the examples do not interfere. The *rehearsal* method requires that at least some portion of the training examples be retained indefinitely. The *pseudo-rehearsal* method overcomes this requirement by using the existing STL network to generate a random set of *pseudoitems* or *virtual examples* that can be rehearsed along along with the new examples.

Robins shows that pseudo-rehearsal is nearly as effective as rehearsal of retained examples. Although the work focuses on the integration of new examples into an STL network, Robins goes on to suggest that pseudo-rehearsal is a potential model for long-term memory consolidation in the mammalian neocortex. He relates this to a seminal neuroscience paper (McClelland et al., 1994) which discusses the complimentary roles of the hippocampus and the neocortex in human learning.

3. Selective knowledge transfer with ηMTL

The above background material has led to a theory of selective knowledge transfer in the context of back-propagation ANNs that can be used to develop a life-long learning system (Silver, 2000). The theory proposes (1) a method of retaining learned task knowledge and recalling it when learning a new task and (2) a method of selectively transferring previously learned knowledge to a new task based on a measure of task relatedness. This section summarizes those aspects of the theory that are concerned with selective transfer using a measure of task relatedness. In the following section we detail those aspects that cover the retention and recall of task knowledge; the major focus of this paper.

3.1 ηMTL: framework for a measure of task relatedness

To optimize the transfer of knowledge within an MTL network, the secondary tasks should be as closely related to the primary task as possible, else negative inductive bias can resultin a less accurate hypothesis. This suggests that tasks have degrees of relatedness to one another and that a measure of relatedness might be used to control the parallel learning of multiple tasks.

Abu-Mostafa develops the mathematics for learning a primary task from *hint* examples of several related tasks, within a single task learning (STL) network (Abu-Mostafa, 1995). Hints are a form of inductive bias that characterize properties of the primary task or task domain such as monotonicity or symmetry of the output with respect to the inputs. The theory states that minimizing the error across all hint examples will contribute toward the development of a more accurate hypothesis for the primary task. We adapt Abu-Mostafa's mathematics to an MTL network with multiple output nodes, one node for each secondary hint task. If a secondary task is related to the primary task, its examples act as hints for training the common feature layer shared by the primary task within the MTL network. If a secondary task is unrelated then its contribution to the MTL network for the primary task can be detrimental and therefore should be minimized.

Consider an objective function to be minimized by the BP algorithm across all task outputs that focuses on the development of the best hypothesis for the primary task:

$$\hat{E} = E_0 + E_1 + ... + E_k + ... + E_t$$

where E_k is the error on the training examples for hint tasks k = 1, ..., t and E_0 is the error on the primary task training examples. By gradient descent the appropriate change to a weight w_{jk} at an output node k is given by $\Delta w_{jk} = -\eta \frac{\partial \hat{E}}{\partial E_k} \frac{\partial E_k}{\partial w_{jk}}$ where η is the learning rate. Under these conditions, $\frac{\partial \hat{E}}{\partial E_k}$, the rate of change of the overall error with respect to the rate of change of the error for task k, can be consider the weight of importance of hint task k for learning the primary task. We define this weight of importance to be the *measure of relatedness*, R_k, between the primary and each of the secondary tasks; that is

$$R_k = \frac{\partial \hat{E}}{\partial E_k} \qquad \text{such that} \qquad \Delta w_{jk} = -\eta R_k \frac{\partial E_k}{\partial w_{jk}} = -\eta_k \frac{\partial E_k}{\partial w_{jk}}$$

Thus, an appropriate measure of relatedness, R_k, for a secondary source task, T_k, must regulate the impact of the task error, E_k, on the formation of shared internal representation.
A separate learning rate, η_k, for each output node k can be considered and kept inside the backward propagated error signal $\frac{\partial E_k}{\partial w_{jk}}$. Thus, by varying η_k it is possible to adjust the amount of weight modification associated with any one task of the network[1]. This modified version of the standard back-propagation learning algorithm for MTL we have called the ηMTL algorithm.
Let the learning rate η_0 for the primary task, T_0, be the full value of the base learning rate η, that is, $\eta_0 = \eta$. Then let R_k vary $0 \leq R_k \leq 1$ for all tasks $k = 1, \ldots, t$, thereby constraining the learning rate for any parallel task to be at most η. Notice, that if $R_k = 1$ for all $k = 0, \ldots, t$, we have standard MTL. Alternatively, if $R_0 = 1$ and $R_k = 0$ for all $k = 1, \ldots, t$, we have standard single task learning (STL) of the primary function. In this way, the ηMTL framework generalizes over STL and MTL.

3.2 The nature of task relatedness

Critical to the transfer of knowledge from a pool of source tasks to a primary task is some measure of relatedness between those tasks (Thrun, 1996, Caruana, 1997, Vilatlta & Drissi, 2002). We define task relatedness in the context of functional transfer: *Let T_k be a secondary task and T_0 a primary task of the same domain with training examples S_k and S_0, respectfully. The* relatedness *of T_k with respect to T_0 in the context of learning system L, that uses functional knowledge transfer, is the utility of using S_k along with S_0 toward the development of an effective hypothesis for T_0. The relatedness of T_k can be expressed as a function of the efficiency and effectiveness of L using S_k to develop a hypothesis for T_0.*
The definition promotes the view that tasks have degrees of relatedness to one and another. Secondary tasks can be partially ordered from most related to least related such that the most related secondary task results in the most effective primary hypothesis, h_0, developed in the shortest period of time.
The definition suggests a brute force approach to determining the relatedness between two tasks. The learning system could learn T_0 in parallel with each secondary task and record the effectiveness of the resulting hypothesis. However, if combinations of secondary tasks are considered then the method would be impractical because of the factorial growth in time complexity. An *a prori* measure of task relatedness is needed.
We have developed and tested several *static, dynamic* and *hybrid* measures of task relatedness that are based on the principals of surface and structural similarity (Silver, 2000, Silver & Mercer, 2001). *Surface similarity* is defined as shallow, easily perceived, external similarity which is a measure of the external functional similarity based on the training

[1] The use of an adaptive or separate learning rate at the node or weight level is not a new concept. It has been used for various purposes, such as (Jacobs, 1988, Naik et al., 1992, Vogl et al., 1988).

examples available for each of the tasks. *Structural similarity* is defined as deep, often complex, internal feature similarity which is the degree to which two developing hypotheses utilize their shared internal representation (particularly, the common feature layer) to produce accurate approximations of tasks.

For the studies presented in this paper we use a hybrid measure that is composed of a static component based on the linear correlation of training set target values and a dynamic component based on the mutual information of hidden node features with respect to training set target values (Silver, 2000).

4. Sequential learning through task rehearsal

4.1 The task rehearsal method.

This section presents the Task Rehearsal Method (TRM) that extends the concept of pseudo-rehearsal (henceforth simply referred to as *rehearsal*) to MTL networks for learning sequences of tasks.

TRM, presented diagrammatically in Figure 3, is a knowledge based inductive learning system that relies on the rehearsal of previously learned tasks when learning a new task within an ηMTL network. After a task T_k has been successfully learned (to a specified level of generalization error), its hypothesis representation is saved in domain knowledge. This representation acts as a surrogate for the space of input-output examples that defines task T_k. Virtual examples of the input-output space for T_k can be produced (with the same level of generalization error) by passing inputs to the domain knowledge representation for T_k and recording the outputs. When learning a new task, T_0, the domain knowledge representations for tasks $T_1 ... T_k ... T_t$ are used to generate corresponding *virtual* output values from the T_0 training examples. The resulting set of virtual examples is used to rehearse the domain knowledge tasks in parallel with the learning of T_0 in a new ηMTL network. The virtual examples can be considered *hints* (as discussed in Section 3.1) that are used to transfer knowledge (provide inductive bias) in a functional manner from $T_1 ... T_k ... T_t$ to T_0.

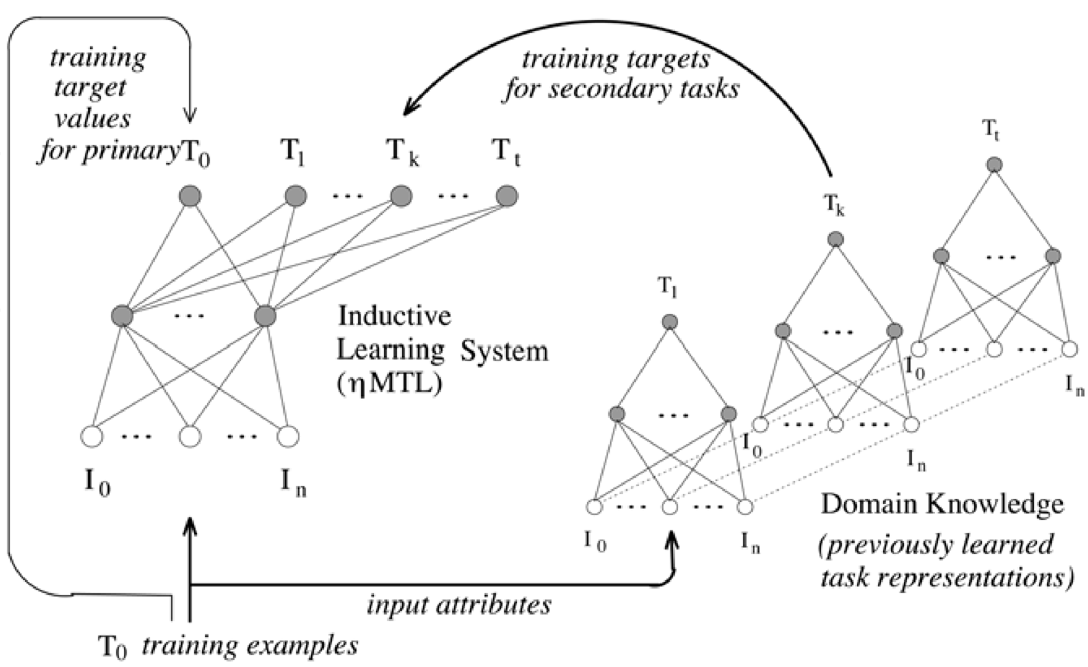


Fig. 3. A model of the Task Rehearsal Method.

The operationalization of TRM sees two sets of feed-forward neural networks interacting during two phases of operation. The following describes the networks and the phases of operation.

The Networks. The set of single output feed-forward networks, labelled *domain knowledge*, is the long-term storage area for the representation of tasks which have been successfully learned. Task representation consists of the network architecture (number of nodes in each layer) and the weights of the connections between the nodes.

The ηMTL back-propagation network, labelled *Inductive Learning System*, can be considered a short-term memory area for learning the new task while being influenced by the inductive bias provided by the domain knowledge networks. The architecture of the ηMTL network must contain enough representation (number of hidden nodes) to develop a sufficiently accurate hypothesis for at least the primary task and potentially for all secondary tasks. There is no requirement that the architecture of the ηMTL network be the same for each new task that is learned.

The major application of TRM is learning a new task from an impoverished training set. In this situation, the lack of training examples has the undesirable side effect of providing insufficient virtual examples for rehearsing accurate hypotheses for the domain knowledge tasks. Consequently, the system cannot provide appropriate inductive bias. To overcome this problem, additional virtual training examples are used with a special *unknown* target classification for the primary task. The back-propagation algorithm used in the Inductive Learning System recognizes these extra training examples and considers their contribution to primary task error to be zero. Consequently, for these examples, only the error of secondary tasks affect the development of the neural network's internal representation.

Phases of Operation. A new task is learned in the ηMTL network during the *knowledge recall and training phase*. Each training example for the primary task T_0 provides n input attributes and a target class value. The target values for secondary tasks $T_1 ... T_k ... T_t$ are, of course, not part of this T_0 training data. These target values must be "recalled" from domain knowledge by feeding the n input attributes into each of the domain knowledge networks and appending the outputs to the original training example. The resulting virtual example is now acceptable to the the ηMTL network for training[2]. Training of the ηMTL network for all tasks begins from random initial weights. The error on a validation set is monitored so as to prevent over-fitting of the data to the network.

The *knowledge retention phase* follows the successful learning of a new task. If the hypothesis for the primary task is able to classify an independent test set of examples with an error rate below a user specified level, the task is considered successfully learned and its representation is saved in domain knowledge. The architectures of the stored networks can be completely different from one another. The fundamental requirement of domain knowledge is an ability to store and retrieve the representations of induced hypotheses and to use these representations to generate accurate virtual examples of the original tasks.

---

[2] The domain knowledge networks output continuous values, therefore, the virtual target values will range between 0 and 1. Continuous target values will more accurately convey the function of the domain knowledge networks and they provide the means by which dichotomous classification tasks may transfer knowledge from related continuous valued tasks in future research.

4.2 The prototype software.

A prototype software system has been developed that implements the TRM shown in Figure 3. The system uses enhanced back-propagation ANN software that is capable of single task learning (STL), MTL or ηMTL. The system employs a batch method of backpropagation of error that utilizes a momentum term in the weight update equation to speed the convergence of the network. A *save-best weights* method is used to save the representation of the network at minimum error on the validation set.

The prototype system uses a sequence table to control the order in which tasks will be learned. For each task in the table, the software moves through the two phases of operation described above. Before learning a new primary task, the examples for the primary task are used to generate the virtual examples for all secondary tasks. A domain knowledge table contains the names of previously learned secondary task representations. If so desired, the table can be populated with names of task representations learned during previous runs of the system.

After a minimum validation error hypothesis has been developed, the TRM software must determine if the hypothesis is sufficiently accurate to be stored within domain knowledge. If the accuracy criteria is met, the hypothesis representation is stored, and the task name is added to the domain knowledge table. If the accuracy criteria is not met, the hypothesis is rejected and no representation or record of the task is kept. A record of the task's name in the domain knowledge table ensures that the associated hypothesis will be considered during the learning of future tasks.

5. Empirical studies

5.1 The domains studied.

This section reports on the testing of TRM against four domains of tasks. The characteristics of an appropriate task domain for testing a life-long learning system is considered in (Silver, 2000). The most important factors are that (1) one or more primary tasks of interest have impoverished sets of training examples insufficient for developing accurate hypotheses under STL, and (2) the domain contain a mix of secondary tasks such that the majority of these tasks are unrelated to the primary task(s) so as to force the life-long learning system to overcome negative inductive bias.

The Band domain consists of seven synthetic tasks. Each task has a band of positive examples across a 2-dimensional input space. The tasks were synthesized so that the primary task T_0 would vary in its relatedness to the other tasks based on the band orientation. A preliminary study showed that T_4, T_5 and T_6 are the more related to T_0 when individually learned in parallel with T_0; they consistently resulted in the most accurate hypotheses for T_0.

The Logic domain consists of eight synthetic tasks. Each positive example is defined by a logical combination of 4 of the 11 input variables of the form, $T_0 : (A > 0.5 \vee B > 0.5) \wedge (C > 0.5 \vee D > 0.5)$. Tasks T_1, T_2 and T_3 are more related to T_0 with T_2 being the most related. The Band and Logic domains have been designed so that all tasks are non-linearly separable; each task requires the use of at least two hidden nodes of a neural network to form an accurate hypothesis.

The coronary artery disease (CAD) domain contains three real medical diagnostic tasks and four synthetic tasks. Each task has five input attributes (age, gender, resting blood pressure, level of chest pain, resting electrocardiogram results). Data for the real tasks were extracted from the heart disease database in the UCI machine learning repository (Detrano, 1989). The data were originally collected from three geographically different locations: the Cleveland Clinic Foundation, Cleveland (*clev*), the Hungarian Institute of Cardiology, Budapest (*hung*) and the V.A. Medical Centre, Long Beach, California (*vamc*). Predicting those patients with disease (50% stenosis of a coronary artery) from the *vamc* hospital is the primary task. Because of the relatively high degree of relatedness between these tasks, data for four additional tasks (*A*, *B*, *C*, and *D*) that vary in their relatedness to the real tasks were synthesized based on knowledge of general rules for predicting CAD.

The forest cover type (Cover) domain consists of six real tasks for predicting the type of tree cover from cartography information. The data concerns the Comanche Peak Wilderness Area of northern Colorado and was downloaded from the UCI repository (Blackard & Dean, 2000). Ten input variables were extracted from the original data (eg. elevation, slope, hillshade at noon, distance to waterway). Each task concerns the prediction of one cover type: Krummholtz, Douglas Fir, Aspen, Ponderosa Pine, Lodgepole Pine or Spruce/Fir. Each task required the use of two or more hidden nodes to produce the best STL models. The challenge is to develop models for Lodgepole Pine and Spruce/Fir from impoverished data sets after having developed models for the first four tasks.

Table 1 summarizes the size of the data sets used for training, validating, and testing each task of each domain under study. The tasks are presented in the order in which they are sequentially learned using TRM. Note that each training set is augmented with additional examples with *unknown* target values for the primary task. The number of additional examples varies for each task so as to ensure there are at least 50, 50, 100 and 126 training examples for the Band, Logic, CAD and Cover domain, respectfully. These additional examples are used by TRM to generate virtual examples for rehearsing the secondary tasks that have been previously learned (see Section 4.1).

Domain	Tasks and Size of Training Set								Val. Set	Test Set
Band	T_1	T_6	T_2	T_5	T_4	T_3	T_0			
	50	35	30	25	20	15	10		20	200
Logic	T_7	T_6	T_5	T_4	T_3	T_2	T_1	T_0		
	50	50	50	45	40	35	30	25	20	200
CAD	A	B	C	D	*clev*	*hung*	*vamc*			
	123	123	123	123	148	30	10		6–64	75–96
Cover	*krum*	*doug*	*aspen*	*pond*	*lodge*	*spruce*				
	126	126	80	70	60	50			64	6000

Table 1. Training, validation, and test set sizes for each task for the four domains.

5.2 Method.

The tasks for each domain are learned in the left-to-right order presented in Table 1 using the TRM system with each of the inductive learning methods: STL, MTL, and ηMTL. The neural networks used are all 3-layer architectures composed of an input layer of as many nodes as input attributes, a hidden layer of sufficient representation for all tasks in the domain and an output layer of as many nodes as there are tasks in the domain. A standard back-propagation learning approach is taken using validation sets to prevent over-fit of the network to the training data. Test sets are used to determine if the hypotheses that are learned are sufficiently accurate to be saved in domain knowledge. We consider an accuracy of 65% to be a minimum level of performance for the domains under study. Therefore, the maximum misclassification rate allowed on a test data set is 35%.

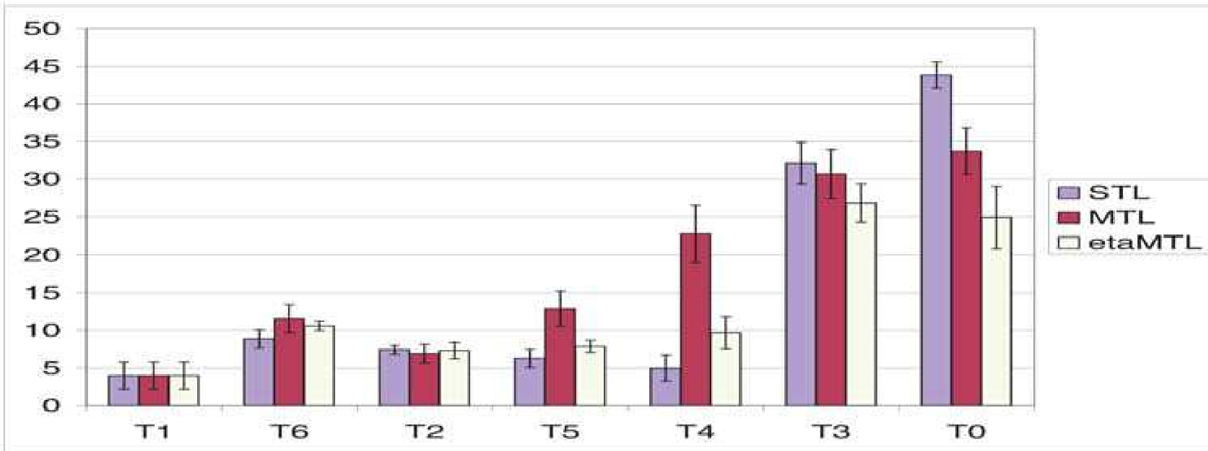
Analysis of the experimental results focuses on the performance of hypotheses developed for each task, particularly those at the end of the learning sequence for each domain. Table 1 shows that training sets for later tasks of each learning sequence have fewer examples than earlier ones. Our goal is to show that the TRM with ηMTL can overcome the impoverished training sets by selectively transferring knowledge from related tasks learned earlier in the sequence and saved in domain knowledge. The performance of the TRM system under each learning method is based on the accuracy of hypotheses against their respective test sets. The mean number of misclassifications from repeated experiments is the measure of performance. We also consider the true positive and true negative proportion statistics for those tasks where the ratio of positive and negative examples are unbalanced. The results shown below are based on 20 repetitions of sequential learning on each domain in which the random initial weights of the networks as well as the training, validation and test examples are resampled.

5.3 Results

Figures 4 and 5 and Table 2 present the test results for hypotheses developed for each task for each domain in the order in which they were learned. The STL results can be used as a baseline for comparison with the TRM results that used either MTL or ηMTL learning. In Table 2 hypotheses developed under MTL and ηMTL with mean percent misclassifications significantly less than STL hypotheses are indicated in bold (95% confidence based on difference of means *t*-test). Hypotheses developed under ηMTL with mean percent misclassifications significantly less than MTL hypotheses are shown in parentheses. The very best results are, therefore, in both bold and parentheses.

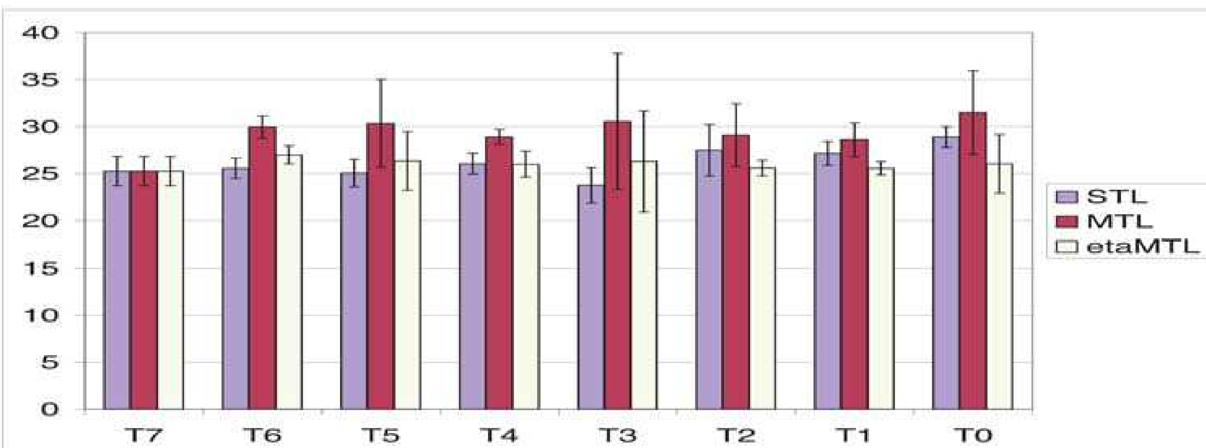
Domain	Mean Percent Misclassifications by Hypotheses							
Band	T_1	T_6	T_2	T_5	T_4	T_3	T_0	
STL	3.95	8.85	7.40	6.25	4.95	32.10	43.80	
MTL	3.95	11.55	6.90	12.85	22.75	30.65	**33.70**	
ηMTL	3.95	10.55	7.30	(7.85)	(9.65)	**26.80**	**(24.90)**	
Logic	T_7	T_6	T_5	T_4	T_3	T_2	T_1	T_0
STL	25.25	25.55	25.05	26.05	23.75	27.50	27.15	28.90
MTL	25.25	29.95	30.35	28.90	30.55	29.10	28.60	31.50
ηMTL	25.25	(27.00)	(26.35)	(26.00)	(26.30)	(25.60)	**(25.55)**	**(26.05)**
CAD	*A*	*B*	*C*	*D*	*clev*	*hung*	*vamc*	
STL	0.53	1.73	7.60	4.27	32.86	23.47	36.35	
MTL	0.53	1.47	5.20	2.40	33.96	**21.10**	**25.21**	
ηMTL	0.53	1.47	4.80	2.27	**(31.43)**	**21.32**	**(20.63)**	
Cover	*krum*	*doug*	*aspen*	*pond*	*lodge*	*spruce*		
STL	8.10	9.10	19.80	13.10	15.10	14.90		
MTL	8.10	8.90	**16.20**	**12.20**	22.20	16.40		
ηMTL	8.10	8.90	**16.00**	**12.10**	(21.20)	16.30		

Table 2. The mean percentage of misclassifications of test set examples by the hypotheses generated by the learning methods under TRM.



(a) Band Domain

(Mean Percentage of Misclassifications).

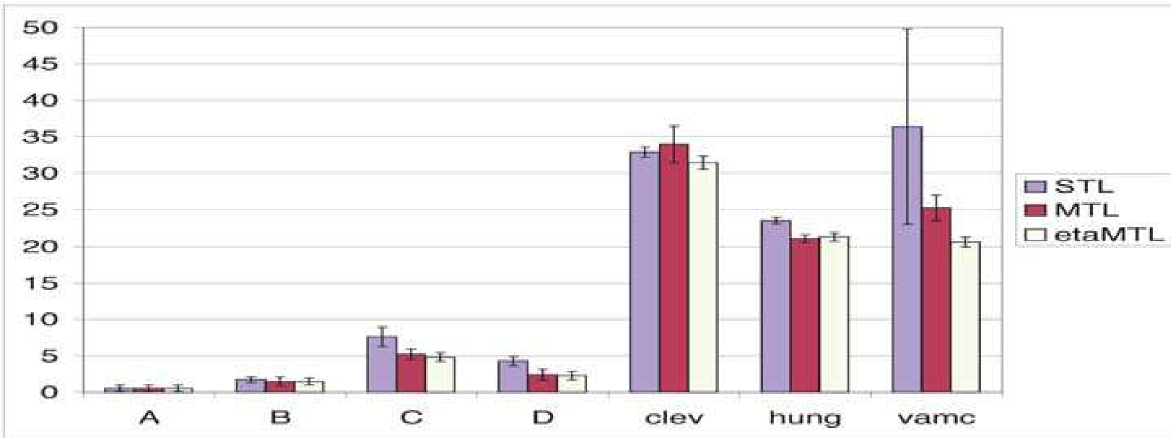


(b) Logic Domain

(Mean Percentage of Misclassifications).

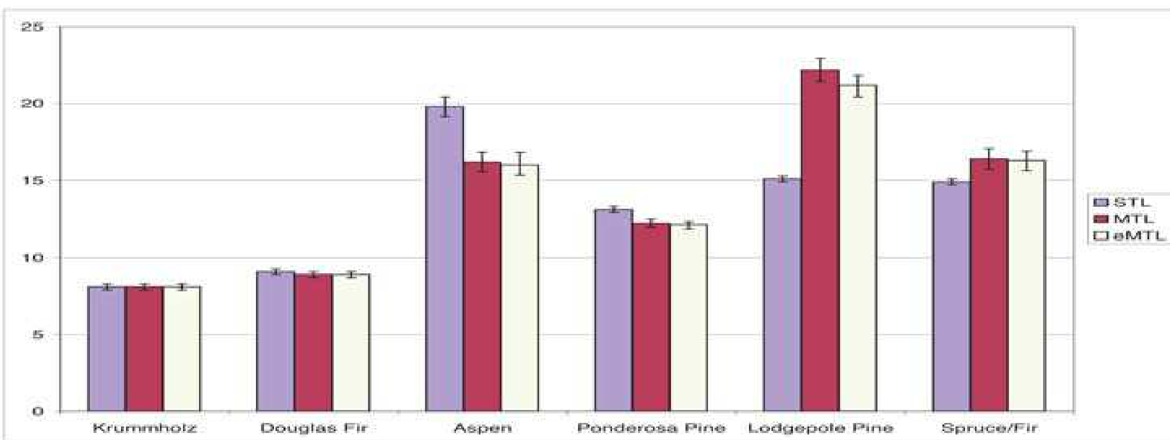
Fig. 4. Performance results from sequential learning on the two synthetic task domains. Shown is the mean percentage of misclassifications by hypotheses generated by each learning method for each task. The results are presented in the order that the tasks were learned.

The results from the synthetic domains (Band, Logic) indicate that hypotheses developed under STL for tasks that have large numbers of training examples (the first four or five tasks) performed as well as or better than hypotheses developed under TRM. Those hypotheses developed under TRM using MTL as the learning method have misclassification rates that are, at times, significantly higher than that of STL hypotheses. The synthetic domains have more unrelated tasks than related ones, therefore, the arbitrary transfer of domain knowledge under this circumstance can have a detrimental effect on learning. This is most evident in the case of learning the Logic domain task T_3, where 40 training examples convey sufficient information to develop relatively accurate hypotheses under STL. Negative inductive bias from unrelated secondary tasks result in MTL hypotheses for T_3 having significantly higher error as compared with STL hypotheses. Inductive bias from secondary hypotheses will always have an effect on the internal representation developed



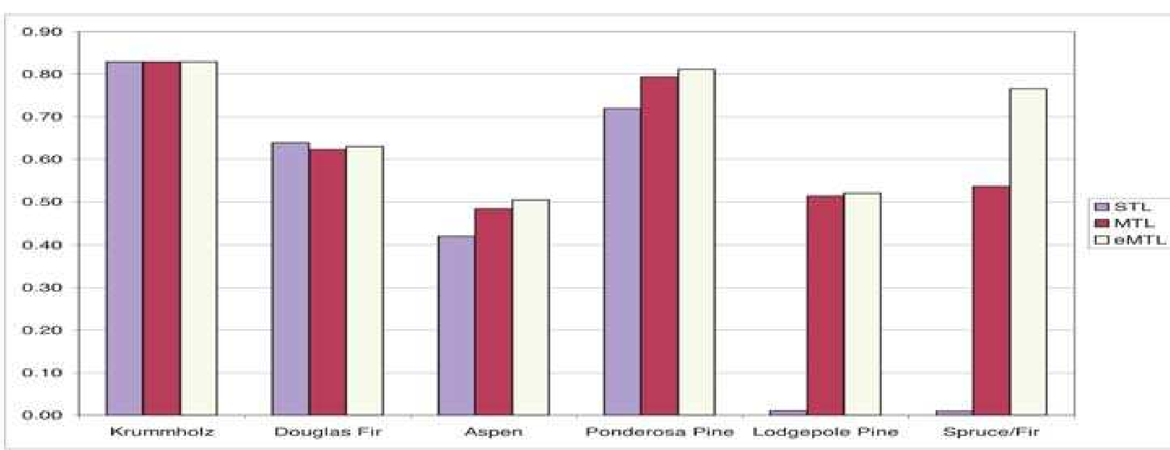
CAD Domain

(Mean Percentage of Misclassifications).



(b) Cover Domain

Domain (Mean Percentage of Misclassifications).



(c) Cover Domain

Domain (True Positive Proportion).

Fig. 5. Performance results from sequential learning on the two real task domains. Shown is the mean percentage of misclassifications by hypotheses generated by each learning method for each task. Also shown in graph (c) is the average true positive proportion for each task of the Cover domain. The results are presented in the order that the tasks were learned.

within the network. The challenge for a knowledge based inductive learning system is to filter out negative bias for the primary task. As shown in Table 2, TRM using ηMTL makes a significant improvement upon the MTL results for the synthetic tasks. The effect of negative inductive bias from unrelated tasks is mitigated by control over the individual learning rates for each of the secondary tasks. The error rate for the first four or five tasks under ηMTL are therefore closer to that of STL.

The results from the real domains (CAD, Cover) indicate that hypotheses developed using TRM with MTL and TRM with ηMTL consistently perform as well as or better than STL hypotheses for the first four or five tasks of each domain. Unlike the synthetic domains, the tasks of the real domains are more closely related to each other. Therefore, the knowledge transferred under MTL from prior tasks is almost always positive. Selective transfer under ηMTL produces only marginally better hypotheses. In interpreting the results of the Cover domain, it should be noted that the mean misclassification percentages for STL hypotheses for the last two tasks of the Cover domain are misleading. As shown by the True Positive Proportion graph of Figure 5, the low misclassification rates on LodgePole Pine and Spruce/Fir are because most of the STL hypotheses developed are naïve, that is, they classify all examples as negative.

The training data for the final two or three tasks of each synthetic and real domain are dramatically impoverished as compared to that of the first task in each learning sequence (see Table 1). STL has difficulty developing accurate hypotheses because the training data for these tasks provides so little information. The TRM with ηMTL augments the impoverished data with positive inductive bias from domain knowledge, resulting in better hypotheses for the last two tasks of all domains. The measure of relatedness reflected in each η_k is able to affect a selective transfer of knowledge from previously learned tasks. The TRM with MTL does not fare as well, particularly on the synthetic domains, because a mix of positive and negative inductive bias occurs from the domain knowledge tasks.

6. Discussion

The following summarizes the key observations that have been made while developing the TRM with ηMTL prototype system and conducting the experiments presented in this paper. A number of avenues for future work are suggested.

6.1 Retention and generation of task knowledge.

The experimental results demonstrate that TRM has the ability to retain accurate task knowledge in the form of neural network representations. The experiments also show that the TRM can generate virtual examples with the same level of accuracy as the retained network hypotheses.

Selective Retention of Accurate Task Knowledge. The TRM prototype system has demonstrated the ability to selectively retain only hypotheses which have met a pre-defined level of generalization accuracy based on the hypotheses classifying independent test sets of examples. This ensures that a level of domain knowledge accuracy is upheld.

Efficiency and Scalability of Task Knowledge Retention. The experiments demonstrated that TRM provides an efficient storage of task knowledge. The hypothesis representations saved in domain knowledge implicitly retain the information from the training examples in a compressed form. In the case of the Logic domain the 8 tasks have a total of 325 actual training examples composed of 12 values each. The total requirement is 3900 units of storage. This same information is retained within eight 11-6-1 network representations composed of 79 weights each. The total storage requirement in this case is 632 units. In general, the number of weights, W, in a 3-layer network can be computed as follows $W = (i+1)h+(h+1)o$; where i, h and o are the number of input, hidden and output nodes, respectfully. When learning a sequence of t tasks, the space requirement for saving all representations within domain knowledge is of $O(t(3+W))$; where 3 is the number of values (i, h, o) required to describe the network architecture. Therefore, the space requirements increase linearly with the number of tasks.

Accuracy and Value of Virtual Examples. The generation of accurate virtual examples from domain knowledge is essential to TRM because they are the means by which accurate knowledge is transferred from a secondary task to the primary hypothesis. The value of a virtual example can be measured by the difference in the mean performance of primary hypotheses developed with and without that example. Supplemental experiments have shown the value of more accurate virtual examples and the incremental value of additional virtual examples when developing hypotheses for related tasks (Silver, 2000). The more accurate the domain knowledge hypotheses (as recorded at the time of learning) the more accurate the virtual examples. This agrees with the reasonable expectation that the effort spent on accurately learning tasks early in life will benefit the learner later in life.

Importance of Input Attribute Range and Resolution. Supplemental experiments have also shown that the TRM can generate virtual examples with the same level of accuracy as the retained hypotheses, provided the input attributes of the virtual examples are within the same range and resolution of values as the examples used to develop the the domain knowledge hypotheses (Silver, 2000). One approach is to record the distribution of the original training and validation examples over the input space. For example the mean value and standard deviation of the original attributes could be computed and saved. These statistics could then be used to generate minimum and maximum boundaries for the input attribute values of virtual examples. Further research is needed into the best choice of attribute values for each new task.

Freedom to use Available Training Examples. TRM provides considerable freedom in the choice of training examples for a new task. Although an MTL method of knowledge transfer is used, there is no need to match specific examples of previously learned tasks. TRM will automatically generate matching virtual target values for the secondary tasks from the input attributes of new task examples. The ability to utilize all available training examples for a new task is a benefit for any life-long learning system.

Abundance of Virtual Examples. The source of inductive bias under TRM is the set of virtual examples generated from domain knowledge for relearning the secondary tasks. The larger the set of virtual examples, the richer the set of functional knowledge from each secondary task. When the number of primary task training examples is small, TRM can generate additional virtual examples for the secondary tasks through the use of primary examples that are marked with the *unknown* target value. These additional virtual examples should be selected bearing in mind the input attribute concerns expressed above. One must also be careful not to overwhelm the information provided in the actual training examples for the primary task with knowledge transferred from the virtual examples of the secondary tasks. Induction must be driven by a fair mix of information from the real examples and bias from the virtual examples. Further research is needed in this area.

Scalability of Virtual Example Generation. Generating virtual examples for TRM requires computational time and space resources. Two options are available. The target values for the secondary tasks can be computed on-line during learning at the cost of increased learning time. Alternatively, the target values can be generated in batch before learning begins at the cost of additional memory. Our current implementation uses the latter approach. The reader may consider that this conflicts with the benefit of efficient storage of training examples. However, because each training example has matching input attribute values, the additional memory cost is only for the storage of the target values of the secondary tasks. As the number of tasks within domain knowledge increases, one solution to this scaling problem is to only generate virtual examples for those domain knowledge tasks that are above a specified threshold of relatedness to the primary task. In this way the space or time complexity for generating the virtual examples would be held constant.

6.2 Selective transfer of task knowledge.

The experimental results show that the TRM with ηMTL can successfully transfer retained task knowledge to the benefit of tasks with impoverished training sets. The results on the last two tasks of each domain are particularly impressive with significantly lower mean number of misclassifications per task than hypotheses produced by either STL or the TRM with MTL.

Tasks having sufficient training examples. STL is able to produce comparatively accurate hypotheses for tasks that have sufficient training examples without needing additional inductive bias. Under MTL a negative transfer of knowledge occurs because of the interference at the common feature layer among unrelated tasks. ηMTL attempts to mitigate this interference but is not able to do so entirely. We are currently working on better measures of task relatedness to reduce the negative bias from unrelated tasks.

Scalability of ηMTL. Because the MTL and ηMTL methods uses the back-propagation algorithm, its time complexity is $O(W_3)$, where W is the number of weights in the ηMTL network (Mitchell, 1997). Naturally, W grows as a function of the number of tasks being learned in parallel. As suggested for the problem of generating virtual examples, a practical solution to lengthy training times is to reduce the number of weights in the network by eliminating secondary tasks that do not meet a predefined level of task relatedness to the primary task. In this way W and the time complexity can be kept constant.

6.3 An alternative explanation for the success of TRM

It could be argued that the reason for the development of more accurate hypotheses under TRM with ηMTL is due to the generation of stochastic noise either by the variation in η_k values during training or because of small errors introduced by the virtual examples for the secondary tasks. A number of authors have shown that stochastic noise can assist the backpropagation algorithm in escaping local minimum to find better hypotheses (Hanson, 1990, Heskes & Kappen, 1993, Wang & Principe, 1999). We have investigated this argument (Silver, 2000) and shown empirically using the Band and Logic domains that stochastic noise in the absence of related tasks cannot produce the same level of positive inductive bias.

6.4 Related work

Task Clustering (TC) Algorithm. Sequential learning, task relatedness and functional knowledge transfer in the context of a memory-based K-nearest neighbour learning system is discussed in (Thrun & O'Sullivan, 1995). The method partitions domain knowledge into clusters of related tasks based on a mutually beneficial Euclidean distance metric used by all tasks within each cluster. When a new task is to be learned, the system estimates the degree of relatedness between the primary task and each task cluster by using that cluster's distance metric to bias the nearest neighbour algorithm. The distance metric that produces the best generalization on the primary task's training examples selects the most related task cluster. The algorithm uses the chosen distance metric to classify future examples of the primary task. Based on experiments using object recognition data, the TC algorithm is shown to construct a meaningful hierarchy of related task clusters based on the distance metric.

In comparison to the TC algorithm, TRM with ηMTL has the advantage of being able to combine inductive bias from one or all domain knowledge tasks and not a predetermined cluster of tasks. The measure of task relatedness allows each new task to select an inductive bias unique to the training examples that are available and the domain knowledge that exists. The TC algorithm also has problems scaling up to a large number of tasks. TRM can overcome this problem by selecting *a priori* only those tasks most closely related to the primary task.

Life-Long Reinforcement Learning. Life-long learning and selective representation transfer is examined in the context of reinforcement learning in (Carroll et al., 2003). Knowledge transfer is recognized as being beneficial only when a source task is sufficiently related to the target task and that a measure of task relatedness is therefore necessary in the presence of multiple source tasks. The research focuses on more efficient learning and does not discuss generalization accuracy.

The TC algorithm, explained above, is extended to reinforcement learning and called the Reinforcement Learning Task Clustering (RLTC) method. RLTC uses a modified Q-learning algorithm that can have its state transition Q-values initialized to that of a related source task or the average of a cluster of related source tasks. Measures of task "similarity" are based on the vector of Q-values for each task. The paper explores methods of determining average Q-values and invariant Q-values within a cluster such that they can be used as the starting point for learning a new and potentially related task. Using a synthetic domain, experiments show that reinforcement tasks can be successfully clustered, and that the Qvalues of task clusters can be used to speed up the learning of a related task or slow down the learning of an unrelated task. The measure of task relatedness allows each new task to select an inductive bias unique to the training examples that are available and the domain knowledge that exists. The authors plan to investigate *a priori* measures of task similarity and the piecewise transfer of knowledge from various tasks in their future work.

Toward Continual Learning. In (Ring, 1997) a neural network based reinforcement learning agent called CHILD is introduced as a first step toward a system that is capable of continual, hierarchical, incremental learning and development. *Continual learning* is defined as "the constant development of increasingly complex behaviours; the process of building more complicated skills on top of those already developed". The system is capable of learning incrementally at each time step and hierarchically using what has been learned in the past to facilitate learning in the future. CHILD can only learn in restricted domains of finite automata tasks but exhibits the seven major characteristics of a continual learner defined in the paper. The system combines Q-learning and temporal transition hierarchies learning and utilizes a novel learning rule to create *higher-order* network units and associated connection weights when needed to overcome network error. Representational knowledge transfer occurs through the use of the weights of previously learned tasks. This can make for rapid learning of related tasks that increase in complexity. Test results on a domain of maze problems demonstrates CHILD's ability to accumulate and use knowledge from related tasks. However, the system shows some difficulty overcoming negative inductive bias from recently learned and unrelated tasks. Catastrophic interference reduces the system's ability to develop an accurate model of a previously learned task.

Knowledge-Based Cascade-Correlation. The Cascade Correlation algorithm (Fahlman and Lebiere, 1990) is extended to the Knowledge Based Cascade Correlation (KBCC) method in (Rivest, 2003, Shultz and Rivest, 2003). KBCC allows previously-learned source networks to compete with each other and with single hidden nodes for recruitment into a target network when learning a new task. The representation of source networks are temporarily connected to the target network and the new connection weights are trained so as to increase the correlation of the source networks' output with the target network's error. The best correlating source network is installed permanently into the target network and its surrounding weights trained to produce a hypothesis for the target task.

Experiments on families of Cartesian input tasks similar to the Band Domain demostrate that the KBCC is able to recruit and use prior task knowledge to quickly develop hypotheses for new tasks. The analysis of the experiments focuses on efficiency of learning and does not examine generalization accuracy. Tasks vary in their shape (rectangles, circles), orientation and size. Although the concept of task "relevance" is discussed and the effect of using different prior tasks on learning is examined the term is not formally defined. A link must be made between the relatedness of prior tasks and the reasons behind recruiting the best correlating prior task. We anticipate that the authors will address these issues in future work.

7. Summary

This paper reviews the importance of inductive bias to learning and discusses the relationship between inductive bias and knowledge transfer. A general model of knowledge-based inductive learning is presented that promotes the retention and use of task knowledge when learning a new task from the same domain. The difference between the representation and functional forms of task knowledge is defined along with the recognition that knowledge retention and transfer can take either form.

An approach to life-long learning is presented in the context of back propagation neural networks. The approach requires (1) a method of selectively transferring previously learned knowledge to a new task and (2) a method of retaining learned task knowledge and its recall. This paper reviews our approach to selective transfer based on a measure of task relatedness, but focuses on the second problem of knowledge retention and recall and presents the Task Rehearsal Method (TRM) as a solution.

TRM, is a knowledge-based inductive learning approach. It is able to retain task knowledge and use that knowledge to bias the induction of hypotheses for new tasks. TRM saves the representation of previously learned neural network hypotheses within a domain knowledge storage area. Domain knowledge is used to generate functional task knowledge in the form of virtual examples at the time of learning a new task. The virtual examples are rehearsed as secondary tasks in parallel with the learning of a new (primary) task using the ηMTL neural network algorithm, a variant of multiple task learning (MTL). The ηMTL algorithm uses a measure of task relatedness to selectively transfer knowledge from the the most related secondary tasks to the hypothesis for a new task. In this way positive inductive bias from prior knowledge is obtained.

The results of repeated experiments on two synthetic and two real domains of tasks demonstrate that TRM with ηMTL produces hypotheses of significantly greater accuracy than either STL or TRM with MTL for tasks with impoverished training data. The success can be attributed to the functional knowledge within the virtual examples generated by the TRM and to the effective use of that knowledge through ηMTL's ability to select the more related secondary tasks. In a similar manner, the TRM with ηMTL is able to mitigate but not eliminate the effect of the negative inductive bias on tasks that have sufficient training examples. Further work on the generation and use of virtual examples as well as the development of better measures of task relatedness are required in order to improve upon selection of inductive bias.

TRM is attractive because of the scalability of its method of task knowledge retention and virtual example generation. Furthermore, TRM provides the freedom to use available training examples for new tasks and the ability to generate additional virtual examples for secondary task rehearsal.

Perhaps of greatest importance, the work on TRM has confirmed the importance and complexity of adding background knowledge to machine learning systems. If inductive bias is a function of stored domain knowledge and a measure of task relatedness, then inductive bias is always relative to the frame of reference created by the previously learned tasks. This requires that domain knowledge be carefully managed; for example, we have observed that redundant task knowledge and task error must be minimized. We are now of the opinion that a consolidated representation of domain knowledge tasks would have a number of benefits over the independent task representations that were used in the experiments of this paper. Consolidated domain knowledge would provide the potential for more efficient learning through representation transfer. It would also allow the use of better measures of task relatedness based on structural similarity of shared representation. More recent research has explored the consolidation of domain knowledge and its use within the TRM (Silver & McCracken, 2003, O'Quinn et al., 2005, Silver & Poirier, 2004).

8. Acknowledgments

The authors wish to express their gratitude for funding provided by the Government of Canada through NSERC.

9. References

Yaser S. Abu-Mostafa. Hints. *Neural Computation*, 7:639–671, 1995.

Jonathan Baxter. Learning internal representations. *Proceedings of the Eighth International Conference on Computational Learning Theory*, 1995.

Jock A. Blackard and Denis J. Dean. Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. *Computers and Electronics in Agriculture*, 24(3):131–151, 2000.

James L. Carroll, Todd Peterson, and Kevin Seppi. Reinforcement learning task clustering. In M. Arif Wani, editor, *Proceedings of the Internation Conference on Machine Learning and Applications (ICMLA'03)*, pages 66–72, Los Angeles, CA, 2003.

Richard A. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

Janosi-A. SteinbrunnW. Pfisterer M. Schmid J. Sandhu S. Guppy K. Lee S. & Froelicher V. Detrano, R. International application of a new probability algorithm for the diagnosis of coronary artery disease. *American Journal of Cardiology*, 64:304–310, 1989.

S.E. Fahlman and C. Lebiere. The cascade-correlation learning architecture. *Advances in Neural Information Processing Systems 2*, 2:524–532, 1990. ed. D.S. Touretsky.

Stephen Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, 11:23–64, 1987.

S. J. Hanson. A stochastic version of the delta rule. *Physica D*, 42:265–272, 1990.

Thomas Heskes and Bert Kappen. On-line learning processes in artificial neural networks. In J. Taylor, editor, *Mathematical Foundations of Neural Networks*. Elsevier, Amsterdam, Netherlands, 1993.

R.A. Jacobs. Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1:295–307, 1988.

E. James Kehoe. A layered network model of associative learning: Learning to learn and configuration. *Psychological Review*, 95(4):411–433, 1988.

James L. McClelland, Bruce L. McNaughton, and Randall C. O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Technical Report PDP.CNS.94.1*, 1994.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: the sequential learning problem. *The Psychology of Learning and Motivation*, 24:109–165, 1989.

R.S. Michalski. Learning = inferencing + memorizing. *Foundations of Knowledge Acquistion: Machine Learning*, pages 1–41, 1993.

Tom Mitchell and Sebastian Thrun. Explanation based neural network learning for robot control. *Advances in Neural Information Processing Systems 5*, 5:287–294, 1993. ed. C. L. Giles and S. J. Hanson and J.D. Cowan.

Tom. M. Mitchell. The need for biases in learning generalizations. *Readings in Machine Learning*, pages 184–191, 1980. ed. Jude W. Shavlik and Thomas G. Dietterich.

Tom M. Mitchell. *Machine Learning*. McGraw Hill, New York, NY, 1997.

D.K. Naik, R. J. Mammone, and A. Agarwal. Meta-neural network approach to learning by learning. *Intelligence Engineering Systems through Artificial Neural Networks*, 2:245–252, 1992.

D.K. Naik and Richard J. Mammone. Learning by learning in neural networks. *Artificial Neural Networks for Speech and Vision*, 1993.

Robert O'Quinn, Daniel L. Silver, and Ryan Poirier. Continued practice and consolidation of a learning task. In *Proceedings of the Meta-Learning Workshop, 22nd International Conference on Machine Learning (ICML 2005)*, Bonn, Germany, 2005.

Lorien Pratt and Barbara Jennings. A survey of transfer between connectionist networks. *Connection Science Special Issue: Transfer in Inductive Systems*, 8(2):163–184, 1996.

Lorien Y. Pratt. Discriminability-based transfer between neural networks. *Advances in Neural Information Processing Systems 5*, 5:204–211, 1993. ed. C. L. Giles and S. J. Hanson and J.D. Cowan.

Mark Ring. Learning sequential tasks by incrementally adding higher orders. *Advances in Neural Information Processing Systems 5*, 5:155–122, 1993. ed. C. L. Giles and S. J. Hanson and J.D. Cowan.

Mark Ring. Child: A first step towards continual learning. *Machine Learning*, 28(1):77–109, 1997.

Francois Rivest. *Knowledge-Transfer in Neural Network: Knowledge-Based Cascade-Correlation*. PhD Thesis, School of Computer Science, McGill University, Montreal, PQ, 2003.

Anthony V. Robins. Catastrophic forgetting, rehearsal, and pseudorehearsal. *Connection Science*, 7:123–146, 1995.

Anthony V. Robins. Consolidation in neural metworks and in the sleeping brain. *Connection Science Special Issue: Transfer in Inductive Systems*, 8(2):259–275, 1996.

Noel E. Sharkey and Amanda J.C. Sharkey. Adaptive generalization and the transfer of knowledge. *Working paper - Center for Connection Science*, 1992.

Jude W. Shavlik and Geoffrey G. Towell. An appraoch to combining explanation-based and neural learning algorithms. *Readings in Machine Learning*, pages 828–839, 1990. ed. Jude W. Shavlik and Thomas G. Dietterich.

Thomas R. Shultz and Francois Rivest. Knowledge-based cascade-correlation: Varying the size and shape of relevant prior knowledge. *New Developments in Psychometrics*, pages 631–638, 2003.

Daniel L. Silver. *Selective Transfer of Neural Network Task Knowledge*. PhD Thesis, Dept. of Computer Science, University of Western Ontario, London, Canada, 2000.

Daniel L. Silver and Peter McCracken. The consolidation of neural network task knowledge. In M. ArifWani, editor, *Proceedings of the International Conference on Machine Learning and Applications (ICMLA'03)*, pages 185–192, Los Angeles, CA, 2003.

Daniel L. Silver and Robert E. Mercer. The parallel transfer of task knowledge using dynamic learning rates based on a measure of relatedness. *Connection Science Special Issue: Transfer in Inductive Systems*, 8(2):277–294, 1996.

Daniel L. Silver and Robert E. Mercer. Selective functional transfer: Inductive bias from related tasks. *Proceedings of the International Conference on Artificial Intelligence and Soft Computing (ASC2001)*, pages 182–189, 2001.

Daniel L. Silver and Ryan Poirier. Sequential consolidation of learned task knowledge. *Lecture Notes in Artificial Intelligence, 17th Conference of the Canadian Society for Computational Studies of Intelligence* (AI'2004), pages 217–232, 2004.

Satinder P. Singh. Transfer of learning by composing solutions for elemental sequential tasks. *Machine Learning*, 1992.

Steven Suddarth and Y Kergoisien. Rule injection hints as a means of improving network performance and learning time. Proceedings of the EURASIP workshop on Neural Networks, 1990.

Sebastian Thrun. Lifelong learning: A case study. *Technical Report CMU-CS-95-208*, Nov 1995.

Sebastian Thrun. Is learning the n^{th} thing any easier than learning the first? *Advances in Neural Information Processing Systems 8*, 8, 1996.

Sebastian Thrun. Lifelong learning algorithms. *Learning to Learn*, pages 181–209, 1997.

Sebastian Thrun and J. O'Sullivan. Clustering learning tasks and the selective cross-task transfer of knowledge. *Technical Report CMU-CS-95-209*, Nov 1995.

Sebastian Thrun and Lorien Pratt. Learning to learn: Introduction and overview. *Learning to Learn*, pages 3–17, 1997.

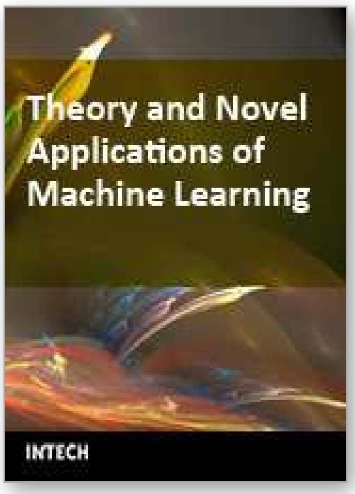
Geoffrey G. Towell, Jude W. Shavlik, and Michiel O. Noordewier. Refinement of approximate domain theories by knowledge-based neural networks. *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90)*, 2:861–866, 1990.

Paul E. Utgoff. *Machine Learning of Inductive Bias*. Kluwer Academc Publisher, Boston, MA, 1986.

Ricardo Vilatlta and Youssef Drissi. A prespective view and survey of meta-learning. *Artificial Intelligence Review*, 18(2):77–95, 2002.

T.P. Vogl, J.K. Mangis, A.K. Rigler, W.T. Zink, and D.L. Alkon. Accelerating the convergence of the back-propagation method. *Biological Cybernetics*, 59:257–263, 1988.

C. Wang and J. C. Principe. Training neural networks with additive noise in the desired signal. *IEEE Transactions on Neural Networks*, 10(6):1511, 1999.



Theory and Novel Applications of Machine Learning
Edited by Meng Joo Er and Yi Zhou

ISBN 978-953-7619-55-4
Hard cover, 376 pages
Publisher InTech
Published online 01, January, 2009
Published in print edition January, 2009

Even since computers were invented, many researchers have been trying to understand how human beings learn and many interesting paradigms and approaches towards emulating human learning abilities have been proposed. The ability of learning is one of the central features of human intelligence, which makes it an important ingredient in both traditional Artificial Intelligence (AI) and emerging Cognitive Science. Machine Learning (ML) draws upon ideas from a diverse set of disciplines, including AI, Probability and Statistics, Computational Complexity, Information Theory, Psychology and Neurobiology, Control Theory and Philosophy. ML involves broad topics including Fuzzy Logic, Neural Networks (NNs), Evolutionary Algorithms (EAs), Probability and Statistics, Decision Trees, etc. Real-world applications of ML are widespread such as Pattern Recognition, Data Mining, Gaming, Bio-science, Telecommunications, Control and Robotics applications. This books reports the latest developments and futuristic trends in ML.