In: Proceedings of the Thirteenth International Conference on Machine Learning
L. Saitta (ed.), Morgan Kaufmann, San Mateo, CA, 1996

Discovering Structure in Multiple Learning Tasks: The TC Algorithm

Sebastian Thrun* **Joseph O'Sullivan**

Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213-3891
http://www.cs.cmu.edu/∼thrun/ or ∼josullvn/

Abstract

Recently, there has been an increased interest in "lifelong" machine learning methods, that transfer knowledge across multiple learning tasks. Such methods have repeatedly been found to outperform conventional, single-task learning algorithms when the learning tasks are appropriately related. To increase robustness of such approaches, methods are desirable that can reason about the relatedness of individual learning tasks, in order to avoid the danger arising from tasks that are unrelated and thus potentially misleading.

This paper describes the task-clustering (TC) algorithm. TC clusters learning tasks into classes of mutually related tasks. When facing a new learning task, TC first determines the most related task cluster, then exploits information selectively from this task cluster only. An empirical study carried out in a mobile robot domain shows that TC outperforms its non-selective counterpart in situations where only a small number of tasks is relevant.

1 INTRODUCTION

One of the exciting new developments in the field of machine learning are algorithms that can gradually improve their ability to learn when applied to a sequence of learning tasks. Motivated by the observation that humans encounter more than just a single learning task during their lifetime, and that they successfully improve their ability to learn [2, 17], several researchers have proposed algorithms that are able to acquire domain-specific knowledge and re-use it in future learning tasks. For example, in the context of face recognition, methods have been developed that improve the recognition accuracy significantly when learning to recognize a face, by transferring face-specific invariances learned in other, previous face recognition tasks [5, 13]. Similar results in the context of object recognition, robot navigation and chess are reported in [26].

*The first author is also affiliated with the University of Bonn, Germany, where part of the research was carried out.

Technically speaking, the problem of learning from multiple tasks can be stated as follows. Given

1. training data for the current learning task (training set),
2. training data for N other, previous learning tasks (called: *support sets*), and
3. a performance measure

find a hypothesis which maximizes the performance in the current (the $N+1$-th) learning task. Notice that item 2 in this list, the data of previous learning tasks (the support sets), does not appear in the usual formulation of machine learning problems. This is because support data might only be indirectly related, *e.g.*, carry different class labels. To utilize this data, mechanisms are required that can acquire and re-use domain-specific knowledge in order to guide the generalization in a knowledgeable way.

To date, there is available a variety of strategies for the transfer of domain-specific knowledge across multiple learning tasks (see [26, 27] for a more detailed survey and comparison):

- learning internal representations for artificial neural networks, *e.g.*, [1, 4, 8, 19, 21, 22, 24],
- learning distance metrics, *e.g.*, [4, 16, 29],
- learning to re-represent the data, *e.g.*, [12, 26, 29],
- learning invariances in classification, *e.g.*, [5, 13, 26, 28],
- learning algorithmic parameters and choosing algorithms, *e.g.*, [6, 20, 25, 30], and
- learning domain models, *e.g.*, [18, 26].

Many of these approaches have been demonstrated empirically to reduce the sample complexity when learning more than one task. However, all of them weigh previous learning tasks equally strongly when transferring knowledge—thus, they may fail when only a small subset of learning tasks is related appropriately. For example, the approaches to object recognition described in [26] generalize better if previous learning tasks involve the recognition of other objects; however—if the learner faced previously unrelated learning tasks (such as stock market prediction), these approaches will most likely fail due to their non-selective nature of their transfer mechanisms. Consequently, it is common practice for human designers to pick a set of tasks which is known to be related appropriately. To overcome this obvious limitation of current approaches, it is desirable to design algorithms that can discover the relation between multiple learning tasks by themselves, and transfer knowledge selectively across *related* learning tasks.

This paper describes such an algorithm, called the TC (task clustering) algorithm. Unlike previous methods, TC transfers knowledge *selectively*, from the most related set of learning tasks only. In order to do so, TC estimates the mutual relatedness between tasks, and builds up an entire *hierarchy* of classes of learning tasks. When a new learning task arrives, TC determines the most related task cluster in the hierarchy of previous learning tasks. Knowledge is transferred selectively from this single cluster only—other task clusters are not employed. The clustering strategy enables TC to handle multiple classes of tasks, each of which may exhibit different characteristics.

To elucidate TC in practice, this paper reports results of a series of experiments carried out in a mobile robot domain [29]. The three key results of this empirical study are:

1. The sample complexity can be reduced significantly when domain-specific knowledge is transferred from previous learning tasks.
2. TC reliably succeeds in partitioning the task space into a (surprisingly) meaningful hierarchy of related tasks.
3. Selective transfer significantly improves the results in cases where only few support tasks are relevant (yet does not hurt the performance when all support tasks are appropriately related).

2 THE TC ALGORITHM

The TC algorithm has been designed to support fast learning of large sets of (binary) classification tasks, that are defined over the same input space. This research has been driven by our interest in fast and data-efficient robot learning algorithms. TC will be introduced in five steps, the first two of which have been adopted from recent literature.

2.1 NEAREST NEIGHBOR GENERALIZATION

At the underlying function approximation level, the TC algorithm uses nearest neighbor for generalization (see e.g. [9, 23]). To determine the proximity of data points, TC uses a *globally weighted Euclidean distance metric*:

$$dist_d(x, y) \quad = \quad \sqrt{\sum_i d^{(i)} \left(x^{(i)} - y^{(i)}\right)^2}$$

Here d denotes an adjustable vector of weighting factors, and the superscript $^{(i)}$ is used to refer to the i-th component of a vector. d parameterizes the space of Euclidean distance metrics. Obviously, d determines the generalization properties of nearest neighbor.

2.2 ADJUSTING THE DISTANCE METRIC

TC transfers knowledge across learning tasks by adjusting d for some tasks, then re-using it in others. Following ideas presented elsewhere [3, 4, 10, 11, 15, 16, 28], this is done by minimizing the distance between training examples that belong to the same class, while maximizing the distance between training examples with opposite class labels:

$$\begin{aligned} E_n(d) \quad &= \quad \sum_{x,y} \delta_{xy}\ dist_d(x, y) \quad \longrightarrow \quad \min \\ \text{where } \delta_{xy} \quad &= \quad \begin{cases} 1 & \text{if } class_n(x) = class_n(y) \\ -1 & \text{if } class_n(x) \neq class_n(y) \end{cases} \end{aligned}$$

Here the subscript n denotes a particular learning task. Let $d^* = \text{argmin}_d\, E(d)$ denote the parameter vector that minimizes E, henceforth called *E-optimal*, and let $dist^*$ be the corresponding optimal distance metric. By minimizing the distance when $class_n(x) = class_n(y)$ and simultaneously maximizing the distance when $class_n(x) \neq class_n(y)$, $dist^*$ focuses on the relevant input dimensions for the n-th learning task. In our implementation, d^* is found using gradient descent.

Notice that d can be optimized simultaneously for multiple learning tasks. Let $A \subset \{1, 2, \ldots N\}$ denote a subset of the support tasks. Then

$$d^*_A \quad = \quad \underset{d}{\text{argmin}} \sum_{n \in A} E_n(d) \tag{1}$$

is the E-optimal parameter vector and $dist^*_A$ the corresponding distance metric for the task set A.

2.3 THE TASK TRANSFER MATRIX

Using the E-optimal distance metric obtained for one task when learning another task is only likely to improve the results when both tasks demand a similar feature weighting. To determine the degree to which tasks are related to each other, TC computes the matrix

$$C \quad = \quad (c_{n,m})$$

which is called the *task transfer matrix*. The task transfer matrix contains a value $c_{n,m}$ for each pair of learning tasks n and m. $c_{n,m}$ is the *expected generalization accuracy obtained in task n when using m's E-optimal distance metric*. Each element $c_{n,m}$ is estimated via k-fold cross-validation, using the E-optimal distance metric of task m and the training set of task n. The task transfer matrix is the basis for clustering tasks and building task hierarchies.

2.4 CLUSTERING TASKS AND THE TASK HIERARCHY

TC clusters all N learning tasks into $T \leq N$ disjunct bins, denoted by $A_1, \ldots, A_T$. This is done by maximizing the following functional:

$$J \quad = \quad \frac{1}{N} \sum_{t=1}^{T} \sum_{n \in A_t} \frac{1}{|A_t|} \sum_{m \in A_t} c_{n,m}$$

J measures the averaged estimated generalization accuracy that is obtained when task $n \in A_t$ uses the E-optimal distance metrics of another task $m \in A_t$ in the same cluster. In other words, maximizing J groups those tasks together that are most related, *i.e.*, those between which transferring

E-optimal distance metrics leads to the largest performance gain.[1] Notice that each of the T resulting task clusters defines a cluster-specific E-optimal distance metric

$$d^*_{A_t} = \operatorname*{argmin}_d \sum_{n \in A_t} E_n(d).$$

which is obtained by minimizing E_{A_t} (*cf.* (1)). By repeating the clustering process for different values of T ($T=1, 2, \ldots N$). Figures 6-8 on page 7 show examples of task hierarchies, which will be explained in more detail in Section 3.

2.5 SELECTIVE TRANSFER TO NOVEL TASKS

When a new learning task arrives, TC identifies the most related task cluster in the hierarchy of previous learning tasks. This is done by minimizing c_{n,A_t} over all task clusters A_t (c_{n,A_t} denotes the task transfer coefficient for using the E-optimal distance metric of task cluster A_t). Notice if an appropriate number of clusters T is unknown, the entire hierarchy is consulted when searching for the most related task cluster. Having determined the most appropriate task cluster, TC uses the E-optimal distance metric $d^*_{A_t}$ of that task cluster for nearest neighbor generalization in the new task.

To summarize, the steps of the TC algorithm are:

1. TC classifies by nearest neighbor, using a globally weighted distance metric.
2. It transfers knowledge across multiple learning tasks by learning a distance metric for some tasks, and re-using it for nearest neighbor generalization in others.
3. To focus the transfer on the most related tasks, TC computes the task transfer matrix, which measures the mutual relation of learning tasks.
4. It constructs the task hierarchy by clustering learning tasks according to the task transfer matrix.
5. When facing a new thing to learn, the distance metric is transferred selectively, from the most related task cluster only.

3 EXPERIMENTAL RESULTS

3.1 SETUP

To evaluate TC thoroughly under a variety of circumstances, it was applied to three different families of binary classification tasks that were obtained using a set of databases shown in Figure 2. The databases were collected with the mobile robot shown in Figure 1, using the color camera (top of the robot) and the 24 sonar proximity sensors (arranged in a ring around the robot) as input. Each database consists of 30 to 200 of snapshots which show examples and counterexamples of a particular concept (such as persons, landmarks, objects, and locations). The first six datasets were constructed with a particular *person* somewhere in front of the robot. Different persons wore different clothes, so that their recognition involved spotting certain colors. The remaining six datasets contained snapshots of a *landmark* (a blue trash-bin), of situations with an *obstacle* in front of the robot, of *doors* (open and closed doors), and finally a collection of random snapshots taken at a particular *location* (lab vs. hallway).

[1]Notice that maximizing J defined over a pairwise matrix (C) is a well-understood combinatorial data clustering problem for which various algorithms exist (see for example [7]).

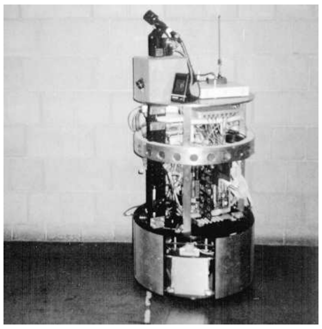

Figure 1: The mobile robot XAVIER is equipped with a camera and 24 sonar sensors.

We defined three families of learning tasks:[2]

- $\mathcal{T}_1$: Task family $\mathcal{T}_1$ consists of thirteen tasks involving the recognition of people, landmarks and locations, using the databases a, b, c, d, g, j, k, and l in Figure 2. The "new" task (testing task) is the task of recognizing the status of doors (open vs. closed). The input space is 324-dimensional: Camera images are subsampled to a 10 by 10 matrix, yielding a total of 300 RGB pixels per image. In addition, the 24 sonar measurements are also presented. Each dataset contains exactly 200 examples.
- $\mathcal{T}_2$: $\mathcal{T}_2$ is aimed to test TC in situations where most tasks are unrelated. It contains three groups of four tasks each: The first group (called 2,4,5,6) consists of four tasks adopted from task family $\mathcal{T}_1$, all involving the recognition of people. The second group (called $2'$,$4'$,$5'$,$6'$) consists of the same four tasks, but here the input pixels are permuted randomly (using the same permutation for all tasks). The third group ($2''$,$4''$,$5''$,$6''$) consists, too, of the same four tasks, but this time the input space of each task is permuted differently (also randomly). The testing task is the same as in task family $\mathcal{T}_1$. Task family $\mathcal{T}_2$, thus, contains only four tasks that are potentially related to the testing tasks. Four other tasks are mutually related but unrelated to the testing task, and four tasks are neither related to the testing task, nor mutually related. As we will see below, non-selective transfer suffers from such unrelated tasks.
- $\mathcal{T}_3$: Task family $\mathcal{T}_3$ consists of nine tasks that correspond to the databases a, d, e, f, h, i, j, k, and l in Figure 2. $\mathcal{T}_3$ uses a more sophisticated input representation. Following the ideas in [14, 15], images in $\mathcal{T}_3$ are encoded using a "view-based" approach to scene recognition. In this approach, feature dimensions are chosen so that large changes in object pose and orientation produce small changes in feature space, yet small changes in

[2]Since our experiments were performed in two stages, not all task families utilized each data set.

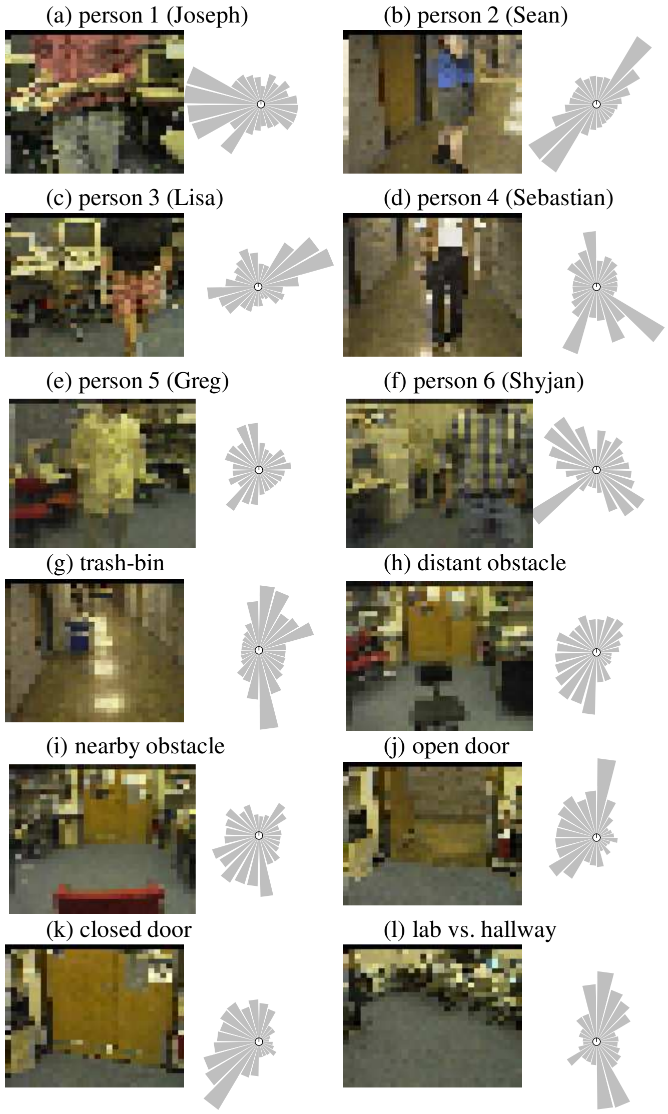

Figure 2: Examples of the learning tasks (image and sonar scan). The actual images are in color.

object "quality" (shape, texture, color) produce relatively large differences in feature space. In particular the representation comprises of 720 features corresponding to color, intensity, texture and correlation,augmented by 24 sonar measurements. The datasets in $\mathcal{T}_3$ were generally smaller; some of them contained as few as 30 examples.

When clustering tasks, each value c_{mn} was estimated using 100-fold cross-validation with a training set of 10 examples per dataset and a testing set of 24 to 190 examples (depending on the dataset). The distance metric (Sect. 2.2) was optimized by gradient descent, which was iterated for 100 steps using a step-size of 0.1 and a momentum of 0.9. Convergence, however, was consistently observed much earlier (often after 6 epochs). The results were not sensitive to these learning parameters. After bounding the distance metric (with $0.01 \leq d^{(i)} \leq 1$), we did not observe noticeable over-fitting, neither for the tasks that the distance metrics were optimized for, nor for the testing task. All experimental results reported below are test set results (*i.e.*, performance was measured for data points that were not part of a training set). They are all averaged over 20 to 100 experiments using different sets of training examples. To illustrate the effect of transfer across tasks, we will compare the E-optimal distance metric (transfer) with a non-optimized (*i.e.*, equally-weighted) distance metric, or, alternatively, with a distance metric that is E-optimal only for the training set. The latter two metrics do not rely on the support sets; thus, there is no transfer. Whenever appropriate, the diagrams also show 95% confidence bars for the true value. All performance graphs show the generalization accuracy (testing set accuracy) for the testing task.

3.2 NON-SELECTIVE TRANSFER

The first question investigated here addresses the effectiveness of learning a distance metric based on support sets (Step 1 and 2 of the TC algorithm, *cf.* Section 2). How much does a learner benefit from a distance metric that has previously been optimized for other, related tasks? We first conducted experiments using (non-selectively) all support tasks for computing the E-optimal distance metric. Non-selective transfer can be understood as a special case of the TC algorithm in which the number of clusters T is set to one.

The first key empirical result of this paper is shown in Figure 3, which compares the accuracy of nearest neighbor as a function of the number of training examples. Both the grey and the thin black curve in Figure 3a illustrate nearest neighbor in the absence of support tasks: The grey curve shows the generalization accuracy of the equally-weighted distance metric, and the thin curve shows the generalization accuracy for the distance metric that is E-optimal for the training set. The thick curve depicts the generalization accuracy when transferring knowledge, using the metric that is E-optimal distance for all 12 support tasks. As can be seen from these graphs, the latter approach shows significantly better results than the other two approaches, particularly in the early phase of learning. This result illustrates the benefit of transferring knowledge across tasks.

There are two ways to quantify these results.

1. **Relative generalization accuracy.** The generalization error is obtained by averaging the curves in Figure 3a. The support set-E-optimal distance metric infers an average classification error of 15.1%, which is only 52.8% of that of the equally-weighted distance metric, and 63.6% of the distance metric that is E-optimal for the training set.
2. **Relative sample complexity.** The second quantity measures the reduction in sample complexity. Figure 3b shows the result of statistical tests on the generalization accuracy for the training set-E-optimal distance metric versus the support set-E-optimal metric, using different numbers of training examples. In the white region,

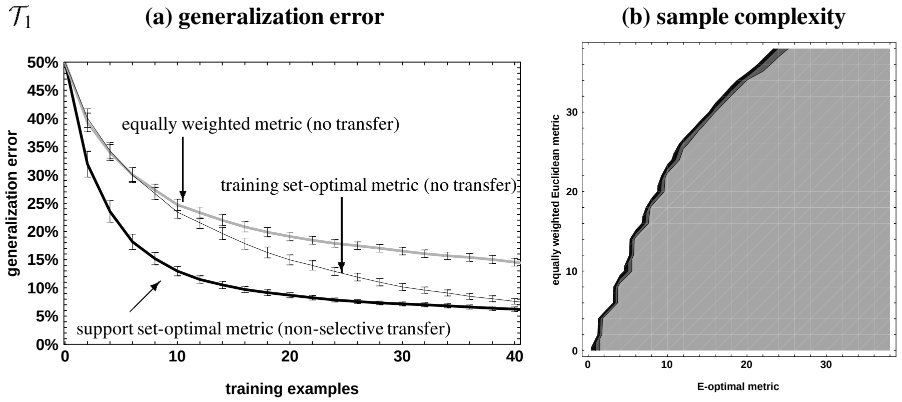

Figure 3: Non-selective transfer in task family $\mathcal{T}_1$. (a) Error as a function of number of training examples. (b) Reduction in sample complexity: Statistical comparison of this error for the best non-transfer metric and the E-optimal distance metric with varying numbers of training examples. In the white (grey) area, the non-transfer (transfer) approach is superior at the 95% confidence level. In the dividing dark region, both methods generalize about equally well. All results shown here are averaged over 100 experiments. The bars indicate 95% confidence intervals.

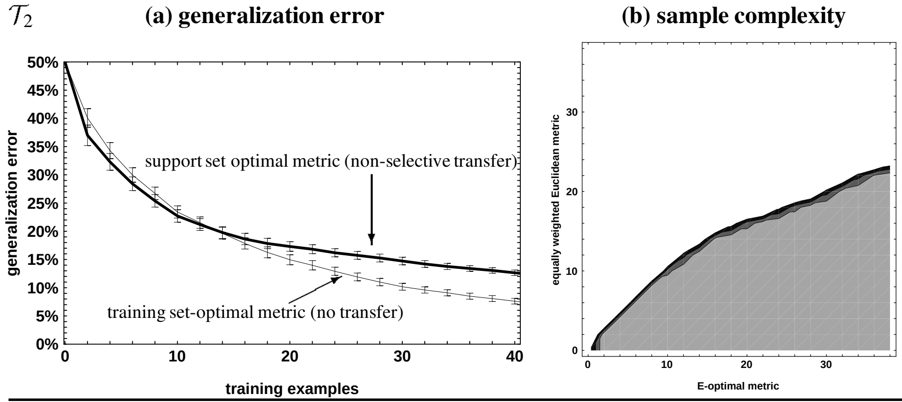

Figure 4: Non-selective transfer in task family $\mathcal{T}_2$. Obviously, unselective transfer *increases* the need for training data in $\mathcal{T}_2$.

the training set-optimal distance metric (no transfer) outperforms the support set-optimal metric (transfer) with at least 95% confidence. In the large grey region, the opposite is the case. In between, both methods work about equally well and their generalization accuracies do not differ significantly (at the 95% level). Notice that, on average, the support set-E-optimal distance metric uses only 55.2% of the number of training examples that are required when using the training-set optimal metric, and 39.2% of the training examples required for the equally-weighted metric. Thus, transfer cuts the sample complexity roughly in half.

To summarize, the generalization error when transferring knowledge is only 63.6% of that inferred by the best non-transfer approach, and it requires only 55.2% of the samples required without transfer. These results apply to task family $\mathcal{T}_1$.

The positive impact of the knowledge transfer depends crucially on the fact that the support tasks are sufficiently related to the testing task. Task family $\mathcal{T}_2$, in which the majority of tasks is unrelated, shows quite the opposite effect. As can be seen in Figure 4, the E-optimal distance metric when transferring knowledge non-selectively is in fact *inferior* to the best non-transfer approach. When transferring knowledge unselectively the average generalization error is 19.7%, which is 9.9% *larger* than that of the best non-transfer approach. The sample complexity *increases* by 18.7% through the (non-selective) transfer of knowledge. These findings support our claim that unselective transfer hurts the performance if the tasks are not appropriately related. As will be shown in the next sections, *selectively* transferring knowledge from the right cluster of tasks can avoid the damaging effects stemming from poorly related tasks.

3.3 CLUSTERING TASKS

Figure 5 shows a normalized version of the transfer matrix $(c_{n,m})$ for each task family. Each row depicts how a particular task n (including the testing task) benefits from knowledge transferred from task m. White boxes indicate that the generalization accuracy of task n improves when the E-optimal distance metric of task m is used instead of the equally-weighted distance metric. Black boxes indicate that the opposite is the case, meaning that tasks are "anti-related." The size of the box visualizes the magnitude of the effect.

In task family $\mathcal{T}_1$, most tasks are either related to the testing task or unrelated, but none of them is notably "anti-related" (first row in Figure 5a). The diagram for the more diverse task family $\mathcal{T}_2$ shows that some of the tasks, in particular $2'$, $2''$, and $4''$, are anti-related to the testing task. In other words, using their respective E-optimal distance metrics will hurt the performance in the testing tasks. However, Figure 5a also shows that the non-permuted tasks 2, 3, 4, and 6 are indeed well-related to the testing task, showing that there exists the opportunity for synergy through knowledge transfer.

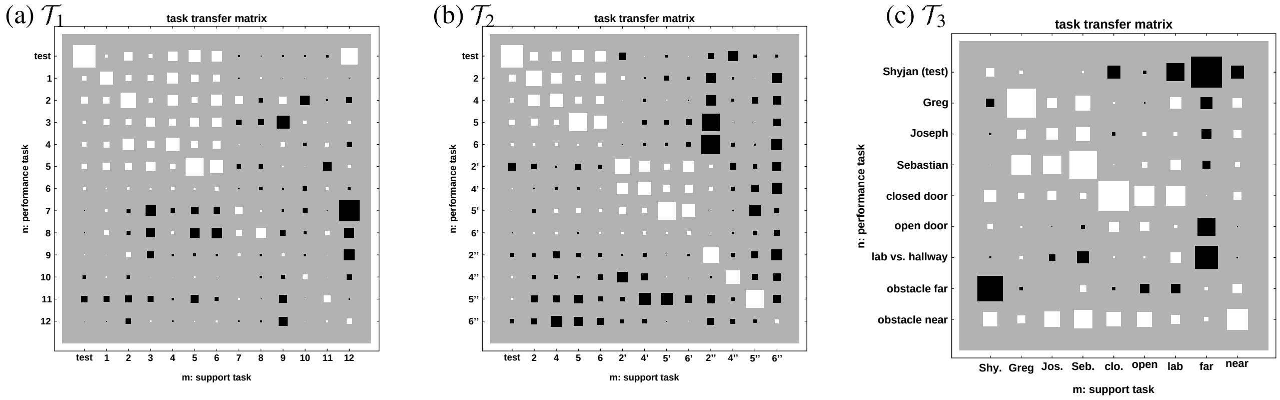

Figure 5: Task transfer matrices $(c_{n,m})$ for the different task families. White values indicate that the error in task n is reduced when the E-optimal distance metric of task m is used. Black values indicate the opposite: tasks are anti-related. The relation of the individual tasks to the testing tasks is also depicted.

3.4 TASK HIERARCHIES

Figures 6-8 depict the task hierarchies for the three task families. These figures illustrate the second key result of the empirical study: In all three task families, TC manages to discover surprisingly meaningful tasks clusters. This is most apparent in task family $\mathcal{T}_3$ (Figure 8). Early on in Figure 8, starting with T=3 clusters, three major clusters have been found, that split the set of tasks into (a) people recognition, (b) determining obstacle proximity, and (c) landmarks/locations. The latter class (c) is then split into one class containing both door-related tasks, and one containing the single location-related task. Notice that the information of the type of learning task has not been communicated explicitly to the TC algorithm—it is discovered from the importance of individual input features the different learning tasks.

Similar results can be found in the hierarchy of task family $\mathcal{T}_2$ (Figure 7). Here the two major task families that use the same encoding ($\{2,4,5,6\}$ and $\{2',4',5',6'\}$) are grouped together. For example, for T=4 partitions TC generates the following task clusters: $\{2,4,5,6\},\{2',4',5',6'\},\{2'',4'',6''\},\{5''\}$. When $T\geq4$ all three different task types are clustered into separate clusters. When T=6, TC groups exactly those tasks together that rely on the same input encoding. Here the clusters are $\{2,4,5,6\},\{2',4',5',6'\},\{2''\},\{4''\},\{5''\}$, and $\{6''\}$.

In task family $\mathcal{T}_1$ (Figure 6), where the differences between different tasks are more subtle, it is interesting to note that the tasks involving the recognition of people form the most similar subgroup (particularly those involving two different people, *cf.* Figure 5a). When $T\geq4$, those tasks that involve the recognition of a person (1 to 9) and than those that do not (10, 11, 12) are always arranged in different clusters. These findings clearly illustrate the second key result of this research: TC indeed manages to find meaningful clusters. In all our experiments, TC discovered the structure that inherently exists for the different tasks.

3.5 SELECTIVE TRANSFER

Figure 9 shows performance results obtained using the TC algorithm in task family $\mathcal{T}_2$, for T=3 clusters (thick black curve in Figure 9a), and when using the entire hierarchy (grey curve). In both experiments, only one of the clusters, namely $\{2,4,5,6\}$, is appropriately related to the testing task, *i.e.*, leads to results that are better than those obtained with the equally-weighted distance function. Across the board, TC selects the best task cluster considerably often, hence generalizes well. For example, when T=4 and only two training examples are given in the test task (one example of an open door, and one of a closed door), TC picks in 59% of our experiments the correct task cluster $\{2,4,5,6\}$. In 24% of all experiments, however, TC selects task cluster $\{5''\}$, in 9% task cluster $\{2',4',5',6'\}$, and in 8% task cluster $\{2'',4'',6''\}$. The situation changes as more training data arrives. With 20 training examples, TC correctly guesses the best task cluster in 91% of all experiments, and with 32 or more patterns it reliably (100%) identifies the best cluster. This illustrates that TC, with some error (when training data is scarce), manages to identify the most relevant tasks.

The performance results obtained in family $\mathcal{T}_2$ illustrate the third key result of the empirical study: Selective transfer is superior to non-selective transfer in situations where many tasks are unrelated (irrelevant). For example, if T=3, TC achieves 14.5% average generalization error in the test task, if knowledge is transferred selectively from the support tasks. Relatively speaking, this is only 73.46% of the average error that is being observed in the non-selective approach (which is 19.7%, *cf.* thick curve in Figure 4a), and it is also considerably close to the best possible distance metric (see [29]). The relative improvement in the sample complexity is even more significant: The sample complexity in the test set is only 58.7% when TC transfers knowledge selectively, when compared to the non-selective counterpart.

When TC is compared to the best non-transfer approach, TC with T=3 uses only 78.5% of the samples to reach the same level of generalization accuracy, and its generalization accuracy is on average 80.8% of that inferred by the

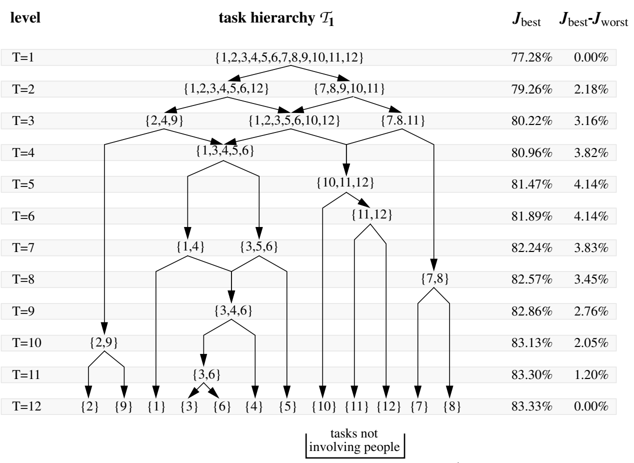

Figure 6: The task hierarchy for task family $\mathcal{T}_1$. The hierarchy is obtained by clustering the task space using different numbers of clusters T. The right part of the diagram depicts the corresponding value for J, and the difference between the best and the worst partitioning.

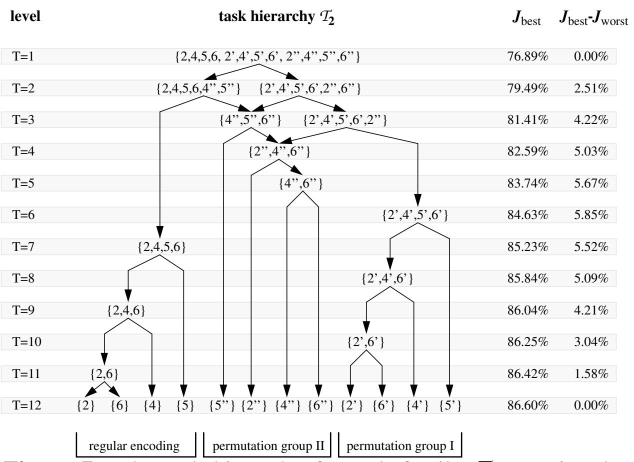

Figure 7: The task hierarchy for task family $\mathcal{T}_2$. Notice that early on, the task hierarchy separates the three different task types. The only related task cluster, $\{2, 4, 5, 6\}$, is identified when $T \geq 3$ clusters are available.

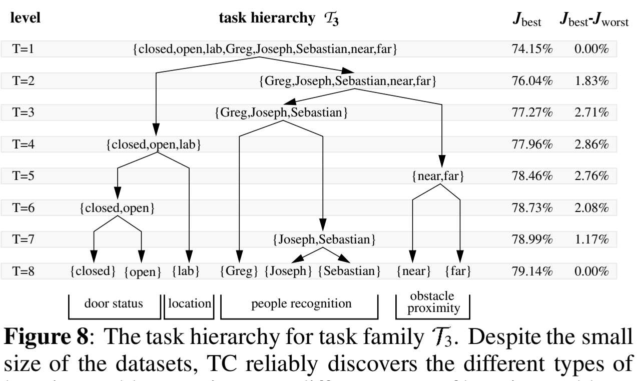

Figure 8: The task hierarchy for task family $\mathcal{T}_3$. Despite the small size of the datasets, TC reliably discovers the different types of learning problems, as it groups different types of learning problems into different branches of the hierarchy.

equally-weighted distance metric. These results are remarkably close to those that could have been achieved if one knew in advance which ones of the 12 support sets were appropriately related. In our experiments, we observed that the number of task clusters T only weakly impacts the results, as long as $T \geq 3$. For smaller values of T, the number of task clusters is insufficient, and TC's performance degrades to that of the regular nearest neighbor with an equally-weighted distance metric.

Figure 10 shows the results obtained when applying TC in task family $\mathcal{T}_3$. These results basically match the results obtained for $\mathcal{T}_1$ and $\mathcal{T}_2$. The most notable difference here is that optimizing the distance metric based on the training set does not lead to an improvement over the equally-weighted, non-optimized distance metric—a finding which we attribute to the fact that the input features in $\mathcal{T}_3$ are more appropriate for image classification tasks (see Section 3.1). Compared to the equally-weighted distance metric, the relative generalization accuracy of TC is 73.1% and the relative sample complexity is 74.3%. Not shown here are results obtained in task family $\mathcal{T}_1$, in which case TC performs approximately as well as its non-selective counterpart (see [29]).

4 DISCUSSION

This paper considers situations in which a learner faces an entire collection of learning tasks. It shows how hierarchical structure can be discovered in the space of learning tasks, and how it can be used to selectively transfer knowledge to other, new learning tasks, in order to boost generalization. The TC algorithm proposed here employs a nearest neighbor algorithm, which transfers knowledge by adjusting the distance metric in some tasks while re-using it in others. To transfer knowledge selectively, TC clusters tasks into bins of related tasks. Relatedness is defined in the context of TC's knowledge transfer mechanisms. When facing a new learning task, TC determines the most related task cluster and selectively transfers knowledge form this one cluster only. In an experimental comparison conducted in a mobile robot perceptual domain it was shown that

1. If tasks are appropriately related, TC's transfer mechanisms successfully reduces the sample complexity. For example, in task family $\mathcal{T}_1$ TC consumes only 55.2% of the training examples that the best non-transfer approach requires.
2. TC's clustering mechanisms manages to discover meaningful task clusters and to build hierarchies of tasks. For example, in task family $\mathcal{T}_3$ TC consistently groups tasks involving people, doors, obstacles, and locations into different bins given that $T \geq 4$ clusters are available. In task family $\mathcal{T}_2$, TC groups the three different task types into separate clusters.
3. If tasks are not appropriately related, selectively transferring knowledge from the most related task cluster improves the results significantly. For example, in task family $\mathcal{T}_2$, in which most tasks are not appropriately related to the testing task, selective transfer requires only

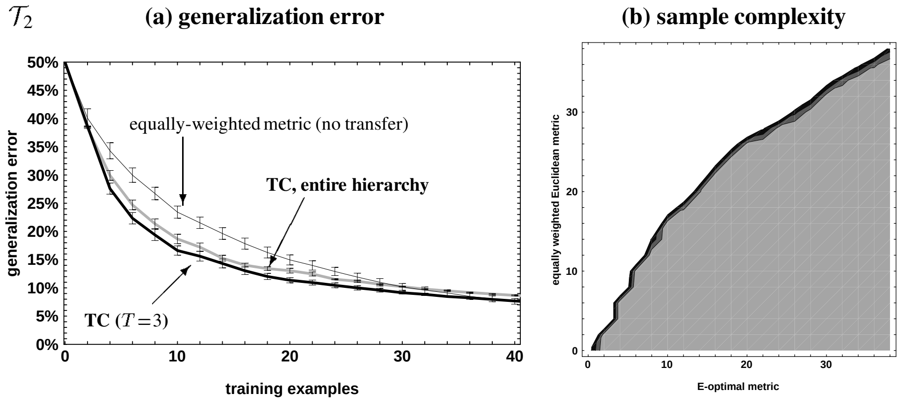

Figure 9: (a) Selective transfer (TC) in task family $\mathcal{T}_2$, using T=3 clusters, and the entire task hierarchy (T=$1 \ldots N$). (b) depicts the statistical comparison of sample complexity for TC with T=3 clusters.

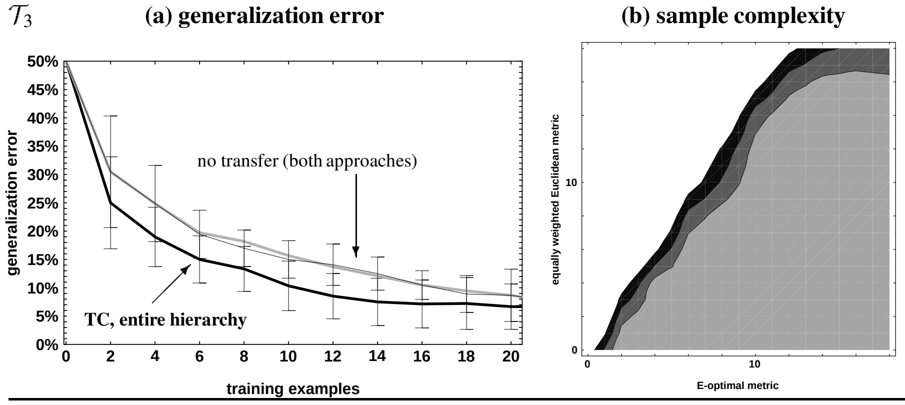

Figure 10: Selective transfer (TC) in task family $\mathcal{T}_3$. The uncertainty in the results (large confidence bars) is due to the the small size of the datasets in task family $\mathcal{T}_3$.

58.7% of the amount of training data required by the corresponding non-selective transfer mechanisms.

These results are well in tune with other results obtained in robot perception, robot control and game playing domains [26], which illustrate that a lifelong learner can generalize more accurately from less data if it transfers knowledge acquired in previous learning tasks.

A key assumption made in the TC approach is the existence of groups of tasks so that all tasks are related within each group. Little is known for cases where the class boundaries are smoother. In such cases, smoother arbitration schemes (*e.g.*, weighting the impact of a task cluster in proportion to $c_{n,m}$) might produce superior results. The results presented in this paper, however, illustrate that even hard class boundaries consistently improve the generalization accuracy. Hard boundaries have the advantage that the cluster-optimal distance metric can be computed off-line, before the arrival of a new learning tasks, which makes TC very fast in practice.

One of the main potential limitations of TC arises from the fact that task clustering is based on pairwise comparisons. TC will not capture effects of transfer that arise if only three or more tasks are involved. It remains to be seen whether pairwise comparisons will prevent TC from finding useful clusters in different application domains. However, a full evaluation of transfer in all subsets of tasks requires time exponentially in the number of tasks N, whereas TC time requirements are quadratic. It even appears feasible to design incremental strategies whose time requirements are in $O(NT)$, which will be more efficient than the current implementation of TC if T is small.

A third limitation of the *current* implementation arises from the fact that the space of all partitions is searched exhaustively (which can only be done when the overall number of tasks is sufficiently small, which was the case in our experiments). Clearly, the complexity of exhaustive search prohibits global optimization for large values of N and T. However, we do not view this as a principal limitation of the TC algorithm, since heuristic and/or stochastic optimization methods are certainly applicable [7]. If learning tasks arrive one after another, task clusters may also be learned *incrementally*, by determining cluster membership when a task arrives. Little is known concerning how much the results presented here depend on the fact that the partitioning always represents the global minimum of J.

The reader may notice that the general scheme underlying the TC approach may be applicable to other approaches that transfer knowledge across multiple learning tasks, such as those surveyed in [26] (see also Section 1). Many of these approaches can learn and transfer more than just a global weighting vector. Of course, for some approaches this will be computationally infeasible, since the general scheme underlying the TC algorithm requires in the order of N^2 comparisons, each involving repeated experiments with transfer across tasks. The key difference of the TC approach to previous approaches lies in TC's ability to transfer knowledge selectively. Rather than weighting all previous learning tasks equally when learning bias for a new one, TC structures the space of learning tasks and reasons about their relatedness. In the light of the experimental findings, we conjecture that the TC approach scales much better application domains in which there are many diverse tasks to be learned, *i.e.*, domains in which the learning tasks are not all just of a single type.

Acknowledgment

The authors wish to thank the anonymous reviewers for their thoughtful comments.

This research is sponsored in part by the National Science Foundation under award IRI-9313367, and by the Wright Laboratory, Aeronautical Systems Center, Air Force Materiel Command, USAF, and the Advanced Research Projects Agency (ARPA) under grant number F33615-93-1-1330. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of NSF, Wright Laboratory or the United States Government.

References

[1] Y. S. Abu-Mostafa. A method for learning from hints. In S. J. Hanson, J. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 73–80, San Mateo, CA, 1993. Morgan Kaufmann.

[2] W.-K. Ahn and W. F. Brewer. Psychological studies of explanation-based learning. In G. DeJong, editor, *Investigating Explanation-Based Learning*. Kluwer Academic Publishers, Boston/Dordrecht/London, 1993.

[3] C. A. Atkeson. Using locally weighted regression for robot learning. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, pages 958–962, Sacramento, CA, 1991.

[4] J. Baxter. *Learning Internal Representations*. PhD thesis, Flinders University, Australia, 1995.

[5] D. Beymer, A. Shashua, and T. Poggio. Example based image analysis and synthesis. A.I. Memo No. 1431, 1993.

[6] C.E. Brodley. *Recursive Automatic Algorithm Selection for Inductive Learning*. PhD thesis, University of Massachusetts, Amherst, MA 01003, 1994.

[7] J. Buhmann. Data clustering and learning. In M. Arbib, editor, *Handbook of Brain Theory and Neural Networks*, pages 278–282. Bradfort Books/MIT Press, 1995.

[8] R. Caruana. Multitask learning: A knowledge-based of source of inductive bias. In P. E. Utgoff, editor, *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48, San Mateo, CA, 1993. Morgan Kaufmann.

[9] R. Franke. Scattered data interpolation: Tests of some methods. *Mathematics of Computation*, 38(157):181–200, 1982.

[10] J. H. Friedman. Flexible metric nearest neighbor classification. 1994.

[11] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. Submitted for publication, 1994.

[12] H. Hild and A. Waibel. Multi-speaker/speaker-independent architectures for the multi-state time delay neural network. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages II 255–258. IEEE, 1993.

[13] M. Lando and S. Edelman. Generalizing from a single view in face recognition. Technical Report CS-TR 95-02, Department of Applied Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot 76100, Israel, 1995.

[14] H. Murase and S. Nayar. Visual learning and recognition of 3-d objects from appearance. *International Journal of Computer Vision*, 14:5–24, 1994.

[15] B. Mel. Seemore: A view-based approach to 3-d object recognition using multiple visual cues. In M.C. Mozer D.S. Touretzky and M.E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*. MIT Press, 1996.

[16] A. W. Moore, D. J. Hill, and M. P. Johnson. An Empirical Investigation of Brute Force to choose Features, Smoothers and Function Approximators. In S. Hanson, S. Judd, and T. Petsche, editors, *Computational Learning Theory and Natural Learning Systems, Volume 3*. MIT Press, 1992.

[17] Y. Moses, S. Ullman, and S. Edelman. Generalization across changes in illumination and viewing position in upright and inverted faces. Technical Report CS-TR 93-14, Department of Applied Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot 76100, Israel, 1993.

[18] J. O'Sullivan, T. M. Mitchell, and S. Thrun. Explanation-based neural network learning from mobile robot perception. In K. Ikeuchi and M. Veloso, editors, *Symbolic Visual Learning*. Oxford University Press, 1996.

[19] L. Y. Pratt. *Transferring Previously Learned Back-Propagation Neural Networks to New Learning Tasks*. PhD thesis, Rutgers University, Department of Computer Science, New Brunswick, NJ 08904, 1993.

[20] L. Rendell, R. Seshu, and D. Tcheng. Layered concept-learning and dynamically-variable bias management. In *Proceedings of IJCAI-87*, pages 308–314, 1987.

[21] N. E. Sharkey and A. J. C. Sharkey. Adaptive generalization and the transfer of knowledge. In *Proceedings of the Second Irish Neural Networks Conference*, Belfast, 1992.

[22] D. Silver and R. Mercer. Toward a model of consolidation: The retention and transfer of neural net task knowledge. In *Proceedings of the INNS World Congress on Neural Networks*, pages 164–169, Volume III, Washington, DC, 1995.

[23] C. Stanfill and D. Waltz. Towards memory-based reasoning. *Communications of the ACM*, 29(12):1213–1228, 1986.

[24] S. C. Suddarth and A. Holden. Symbolic neural systems and the use of hints for developing complex systems. *International Journal of Machine Studies*, 35, 1991.

[25] R. S. Sutton. Adapting bias by gradient descent: An incremental version of delta-bar-delta. In *Proceeding of Tenth National Conference on Artificial Intelligence AAAI-92*, pages 171–176, Menlo Park, CA, July 1992. AAAI, AAAI Press/The MIT Press.

[26] S. Thrun. *Explanation-Based Neural Network Learning: A Lifelong Learning Approach*. Kluwer Academic Publishers, Boston, MA, 1996. to appear.

[27] S. Thrun. Is learning the n-th thing any easier than learning the first? In *Advances in Neural Information Processing Systems 8*, MIT Press, 1996.

[28] S. Thrun and T. M. Mitchell. Learning one more thing. In *Proceedings of IJCAI-95*, 1995. Also appeared as CMU Technical Report CMU-CS-94-184, 1994.

[29] S. Thrun and J. O'Sullivan. Clustering learning tasks and the selective cross-task transfer of knowledge. Technical Report CMU-CS-95-209, Carnegie Mellon University, School of Computer Science, Pittsburgh, PA 15213, 1995.

[30] P. E. Utgoff. *Machine Learning of Inductive Bias*. Kluwer Academic Publishers, 1986.