Learning One More Thing

Sebastian Thrun
Universität Bonn
Institut für Informatik III
Römerstr. 164, D–53117 Bonn, Germany

Tom M. Mitchell
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890, USA

Abstract

Most research on machine learning has focused on scenarios in which a learner faces a single, isolated learning task. The lifelong learning framework assumes that the learner encounters a multitude of related learning tasks over its lifetime, providing the opportunity for the transfer of knowledge among these. This paper studies lifelong learning in the context of binary classification. It presents the *invariance approach*, in which knowledge is transferred via a learned model of the invariances of the domain. Results on learning to recognize objects from color images demonstrate superior generalization capabilities if invariances are learned and used to bias subsequent learning.

1 Introduction

Supervised learning is concerned with learning an unknown target function from a finite collection of input-output examples of that function. Formally, the framework of supervised learning can be characterized as follows. Let F denote the set of all target functions. For example, in a robot arm domain, F might be the set of all kinematic functions for robots with three joints. Every function $f \in F$ is defined over the same input space, denoted by I, and the same output space, denoted by O. The learner has a set of hypotheses that it can consider, denoted by H, which might or might not be different from F. For example, the set H could be the set of all artificial neural networks with 20 hidden units, or, alternatively, the set of all decision trees with depth less than 10. Throughout this paper, we make the simplifying assumption that all functions in F are binary classifiers, *i.e.*, $O = \{0, 1\}$. We will refer to instances that fall into class 1 as positive instances, and to those that fall into class 0 as negative instances.

To learn an unknown target function $f^* \in F$, the learner is given a finite collection of input-output examples (training examples)

$$X = \{\langle i, f^*(i)\rangle\}, \tag{1}$$

which are possibly distorted by noise. The goal of the learner is to generate a hypothesis $h \in H$, such that the deviation (error)

$$E \quad = \quad \sum_{i/inI} Prob(i) \ \ ||f^*(i) - h(i)|| \tag{2}$$

between the target function f^* and h on future examples will be as small as possible. Here $Prob$ is the probability distribution according to which the training examples are generated. $Prob$ is generally unknown to the learner, as is f^*.

Standard supervised learning focuses on learning a single target function f^*, and training data is assumed to be available only for this one function. However, if functions in F are appropriately related, it can be helpful to have access to training examples of other functions f in F as well. For example, consider a robot whose task is to find and fetch various objects, using its camera for object recognition. Let F be the set of recognition (*i.e.*, classification) functions for all objects, one for each potential target object, and let the target function $f^* \in F$ correspond to an object the robot must learn to recognize. X, the training set, will consist of positive and negative examples of this object. The task of the learner is to find an h which minimizes E. In particular, the robot should learn to recognize the target object invariant of rotation, translation, scaling in size, change of lighting and so on. Intuitively speaking, the more profound the learner's initial understanding of these invariances, the fewer training examples it will require for reliable learning. Because these invariances are common to all functions in F, images showing other objects can provide additional information and hence support learning f^*.

This example illustrates the idea of lifelong learning. In lifelong learning, a collection of related learning problems is encountered over the lifetime of the learner. When learning the n-th task, the learner may employ knowledge gathered in the previous $n-1$ tasks to improve its performance [Thrun and Mitchell, to appear].

This paper considers a particular form of lifelong learning in which the learning tasks correspond to learning boolean classifications (concepts), and in which the previous experience consists of training examples of other classification functions from the same family F. More formally, in addition to the set of training examples X for the target function f^*, the learner is also provided $n-1$ sets of examples

$$\begin{aligned} X_k \quad = \quad \{\langle i, f_k(i)\rangle\} \qquad & (k \in \{k_1, k_2, \ldots, k_{n-1}\} \\ & \text{with } \ k_j \in \{1, 2, \ldots, |F|\} \\ & \forall j \in \{1, 2, \ldots, n-1\}) \end{aligned} \tag{3}$$

of other functions $\{f_{k_1}, f_{k_2}, \ldots, f_{k_{n-1}}\} \subset F$ taken from the same function family F. Since this additional data can support learning f^*, we shall call each X_k a *support set* for X. The set

Given:

- a space of hypotheses $H : I \longrightarrow O$
- a set of training examples $X = \{\langle i, f^*(i)\rangle\}$ of some unknown target function $f^* \in F$, drawn with probability distribution $Prob$.
- in lifelong supervised learning: a collection of support sets $Y = \{X_k\}$, which characterize other functions $f_k \in F$. Here $X_k = \{\langle i, f_k(i)\rangle\}$.

Determine:

a hypothesis $h \in H$ that minimizes

$$\sum_{i \in I} Prob(i)\ ||f^*(i) - h(i)||$$

Table 1: Standard and lifelong supervised learning.

of available support sets for X, $\{X_k | k = k_1, k_2, \ldots, k_{n-1}\}$, will be denoted by Y. Notice that the input-output examples in the support sets X_k may have been drawn from different probability distributions.

Support sets can be useful in a variety of real-world scenarios. For example, Sinha [Sinha, 1994] and Lando/Edelman [Lando and Edelman, 1995] have proposed approaches that improve the recognition rate of human faces based on knowledge learned by analyzing different views of other, related faces. In speaker-dependent approaches to speech recognition, learning to recognize personal speech is often done by speaker adaptation methods. Speaker adaptation simplifies the learning task by using knowledge learned from other, similar speakers (*e.g.*, see [Hild and Waibel, 1993]). Other approaches that use related functions to change the bias of an inductive learner can be found in [Utgoff, 1986], [Rendell *et al.*, 1987], [Suddarth and Kergosien, 1990], [Moore *et al.*, 1992], [Sutton, 1992], [Caruana, 1993], and [Pratt, 1993].

Table 1 summarizes the problem definitions of standard supervised learning and the lifelong supervised learning problem. In lifelong supervised learning, the learner is given a collection Y of support sets, in addition to the training set X and the hypothesis space H. This raises two fundamental questions:

1. How can a learner use support sets to generalize more accurately?
2. Under what conditions will a learner benefit from support sets?

This paper does not provide general answers to these questions. Instead, it proposes one particular approach, namely learning invariance functions, which relies on certain assumptions regarding the function set F. It also presents empirical evidence that this approach to using support sets can significantly improve generalization accuracy when learning to recognize objects based on visual data.

2 The Invariance Approach

The invariance approach first learns an invariance function σ from the support sets in Y. This function is then used to bias the learner as it selects a hypothesis to fit the training examples X of the target function f^*.

2.1 Invariance Functions

Let $Y = \{X_k\}$ be a collection of support sets for learning f^*. Recall our assumption that all functions in F have binary output values. Hence, each example in a support set is either positive (i.e., output 1) or negative (i.e., output 0). Consider a target function, $f_k \in F$ with $k \in \{1, \ldots, |F|\}$, and a pair of examples, say $i \in I$ and $j \in I$. A local *invariance operator* $\tau_k : I \times I \longrightarrow \{0, 1\}$ is a mapping from a pair of input vectors defined as follows:

$$\tau_k(i,j) \quad = \quad \begin{cases} 1 & \text{if } f_k(i) = f_k(j) = 1 \\ 0 & \text{if } f_k(i) \neq f_k(j) \\ \text{not defined} & \text{if } f_k(i) = f_k(j) = 0 \end{cases}$$

The local invariance operator indicates whether both instances are members of class 1 (positive examples) relative to f_k. If $\tau_k(i,j) = 1$, then f_k is invariant with respect to the difference between i and j. Notice that positive and negative instances of f_k are not treated symmetrically in the definition of τ.

The local invariance operators τ_k ($k = 1, \ldots, |F|$) define a (global) *invariance function* for F, denoted by $\sigma : I \times I \longrightarrow \{0, 1\}$. For two examples, i and j, $\sigma(i,j)$ is 1 if there exists a k for which $\tau_k(i,j) = 1$. Likewise, $\sigma(i,j)$ is 0 if there exists a k for which $\tau_k(i,j) = 0$:

$$\sigma(i,j) \quad = \quad \begin{cases} 1, & \text{if } \exists k \in \{1, \ldots, |F|\} \text{ with } \tau_k(i,j) = 1 \\ 0, & \text{if } \exists k \in \{1, \ldots, |F|\} \text{ with } \tau_k(i,j) = 0 \\ \text{not defined}, & \text{otherwise} \end{cases}$$

The invariance functionσ behaves like an invariance operator, but it does not depend on k. It is important to notice that the invariance function can be ill-defined. This is the case if there exist two examples which both belong to class 1 under one target function, but which belong to different classes under a second target function:

$$\exists i, j \in I, k, k' \in \{1, \ldots, |F|\} : \tau_k(i,j) = 1 \ \wedge \tau_{k'}(i,j) = 0$$

In such cases the invariance mapping is ambiguous and is not even a mathematical function. A class of functions F is said to obey the *invariance property* if its invariance function is non-ambiguous[1]. The invariance property is a central assumption for the invariance approach to lifelong classification learning.

The concept of invariance functions is quite powerful. Suppose F holds the invariance property. If σ is known, every training instance i for an arbitrary function $f_k \in F$ can be correctly classified, given there is at least one positive instance of f_k available. To see, assume $i_{\text{pos}} \in I$ is known to be a positive instance for f_k. Then for any instance $i \in I$, $\sigma(i, i_{\text{pos}})$ will be 1 if and only if $f_k(i) = 1$. Although the invariance property imposes a restriction on the function family F, it holds true for quite a few real-world problems, such as those typically studied in character recognition, speech understanding, and various other domains. For example, a function family obeys the invariance property if all positive classes (of all functions f_k) are disjoint. One such function family is the family of object recognition functions defined over distinct objects.

2.2 Learning the Invariants

In the lifelong learning regime studied in this paper, σ is not given. However, an approximation to σ, denoted by $\hat{\sigma}$ can

[1] It is generally acceptable for the invariance function to be ambiguous, as long as the probability for generating ambiguously classified pairs of examples is zero.

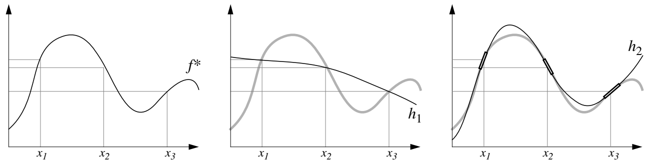

Figure 1: **Fitting values and slopes:** Let f^* be the target function for which three examples $\langle x_1, f^*(x_1)\rangle$, $\langle x_2, f^*(x_2)\rangle$, and $\langle x_3, f^*(x_3)\rangle$ are known. Based on these points the learner might generate the hypothesis h_1. If the slopes are also known, the learner can do much better: h_2.

be learned. Since σ does not depend upon the specific target function f^*, every support set $X_k \in Y$ can be used to train $\hat{\sigma}$, as long as there is at least one positive instance available in X_k. For all $k \in \{1, \ldots, |Y|\}$, training examples for $\hat{\sigma}$ are constructed from examples $i, j \in X_k$:

$$\langle (i, j), \tau_k(i, j)\rangle$$

Here τ_k must be defined, i.e., at least one of the examples i and j must be positive under f_k. In the experiments described below, σ is approximated by training an artificial neural network using the Backpropagation algorithm.

The invariance network, once learned, can be used in conjunction with a training set X to infer values for f^*. Let $X_{\text{pos}} \subset X$ be the set of positive training examples in X. Then for any i_{pos} in X_{pos}, $\hat{\sigma}(i, i_{\text{pos}})$ estimates $f^*(i)$ for $i \in I$. If this estimate is interpreted as a probability (for the event that i is positive under f^*), Bayes' rule can be applied

$$Prob(f^*(i){=}1) \;=\; 1 - \left(1 + \prod_{i_{\text{pos}} \in X_{\text{pos}}} \frac{\hat{\sigma}(i, i_{\text{pos}})}{1 - \hat{\sigma}(i, i_{\text{pos}})}\right)^{-1} \quad (4)$$

Notice that in this approach, $\hat{\sigma}$ is similar to a distance metric that is obtained from the support sets [Moore *et al.*, 1992; Baxter, 1995]. The invariance network $\hat{\sigma}$ generalizes the notion of a distance metric, because the triangle inequality need not hold, and because an instance i_{pos} can provide evidence that i is member of the opposite class (iff $\hat{\sigma}(i, i_{\text{pos}}) < 0.5$).

In general $\hat{\sigma}$ might not be accurate enough to describe f^* correctly. This may be because of modeling limitations, noise, or lack of training data. We will therefore describe an alternative approach to the lifelong learning problem that employs the invariance network, which has been found empirically to generalize more accurately.

2.3 Extracting Slopes to Guide Generalization

The remainder of this section describes a hybrid neural network learning algorithm for learning f^*. This algorithm is a special case of both the Tangent-Prop algorithm [Simard *et al.*, 1992] and the explanation-based neural network learning (EBNN) algorithm [Mitchell and Thrun, 1993]. Here we will refer to it as EBNN.

Suppose we are given a training set X, and an invariance network $\hat{\sigma}$ that has been trained using a collection of support sets Y. We are now interested in learning f^*. One could, of course, ignore the invariance network and the support sets altogether and train a neural network purely based on the training data X. The training set X imposes a collection of constraints on the output values for the hypothesis h. If h is represented by an artificial neural network, as is the case in the experiments reported below, the Backpropagation (BP) algorithm can be used to fit X.

EBNN does this, but it also derives additional constraints using the invariance network. More precisely, in addition to the value constraints in X, EBNN derives constraints on the *slopes* (tangents) for the hypothesis h. To see how this is done, consider a training example i, taken from the training set X. Let i_{pos} be an arbitrary *positive* example in X. Then, $\hat{\sigma}(i, i_{\text{pos}})$ determines whether i and i_{pos} belong to the same class—information that is readily available, since we are given the classes of i and i_{pos}. However, predicting the class using the invariance network also allows us to determine the output-input slopes of the invariance network. These slopes measure the sensitivity of class membership with respect to the input features in i. This is done by computing the partial derivative of $\hat{\sigma}$ with respect to i at (i, i_{pos}):

$$\nabla_i \hat{\sigma}(i) \;:=\; \frac{\partial \hat{\sigma}(i, i_{\text{pos}})}{\partial i}$$

$\nabla_i \hat{\sigma}(i)$ measures how infinitesimal changes in i will affect the classification of i. Since $\hat{\sigma}(\cdot, i_{\text{pos}})$ is an approximation to f^*, $\nabla_i \hat{\sigma}(i)$ approximates the slope $\nabla_i f^*(i)$. Consequently, instead of fitting training examples of the type $\langle i, f^*(i)\rangle$, EBNN fits training examples of the type

$$\langle i, f^*(i), \nabla_i f^*(i)\rangle.$$

Gradient descent can be used to fit training examples of this type, as explained in [Simard *et al.*, 1992]. Fig. 1 illustrates the utility of this additional slope information in function fitting.

Notice if multiple positive instances are available in X, slopes can be derived from each one. In this case, averaged slopes are used to constrain the target function:

$$\nabla_i \hat{\sigma}(i) \;:=\; \frac{1}{|X_{\text{pos}}|} \sum_{i_{\text{pos}} \in X_{\text{pos}}} \frac{\partial \hat{\sigma}(i, i_{\text{pos}})}{\partial i} \quad (5)$$

Here $X_{\text{pos}} \subset X$ denotes the set of positive examples in X. The application of the EBNN algorithm to learning with invariance networks is summarized in Table 2.

1. *Let $X_{\text{pos}} \subset X$ be the set of positive training examples in X.*
2. *Let $X' = \emptyset$*
3. *For each training example $\langle i, f^*(i)\rangle \in X_{\text{pos}}$ do:*
   (a) *Compute $\nabla_i \hat{\sigma}(i) = \frac{1}{|X_{\text{pos}}|} \sum_{i_{\text{pos}} \in X_{\text{pos}}} \frac{\partial \hat{\sigma}(i)(i_{\text{pos}})}{\partial i}$ using the invariance network $\hat{\sigma}$.*
   (b) *Let $X' = X' + \langle i, f^*(i), \nabla_i \hat{\sigma}(i)\rangle$*
4. *Fit X'.*

Table 2: Application of EBNN to learning with invariance networks.

Generally speaking, slope information extracted from the invariance network is a linear approximation to the variances and invariances of F at a specific point in I. Along the invariant directions slopes will be approximately zero, while along others they will be large. For example, in the aforementioned

Figure 2: Objects (left) and corresponding network inputs (right). A hundred images of a bottle, a hat, a hammer, a coke can, and a book were used to train and test the invariance network. Afterwards, the classification network was trained to distinguish the shoe from the glasses.

find-and-fetch tasks, it might happen that color is an important feature for classification while brightness is not. This is typically the case in situations with changing illumination. In this case, the invariance network could learn to ignore brightness, and hence the slopes of its classification with respect to brightness would be approximately zero. The slopes for color, however, would be large, given that slight color changes imply that the object would belong to a different class.

When training the classification network, slopes provide additional information about the sensitivity of the target function with respect to its input features. Hence, the invariance network can be said to bias the learning of the classification network. However, since EBNN trains on both slopes and values simultaneously, errors in this bias (incorrect slopes due to approximations in the learned invariance network) can be overturned by the observed training example values in X. The robustness of EBNN to errors in estimated slopes has been verified empirically in a robot navigation domain [Mitchell and Thrun, 1993].

3 Example

3.1 The Domain: Object Recognition

To illustrate the transfer of knowledge via the invariance network, we collected a database of 700 color camera images of seven different objects (100 images per object), as depicted in Fig. 2 (left columns).

Object	color	size
bottle	green	medium
hat	blue and white	large
hammer	brown and black	medium
can	red	medium
book	yellow	depending on perspective
shoe	brown	medium
glasses	black	small

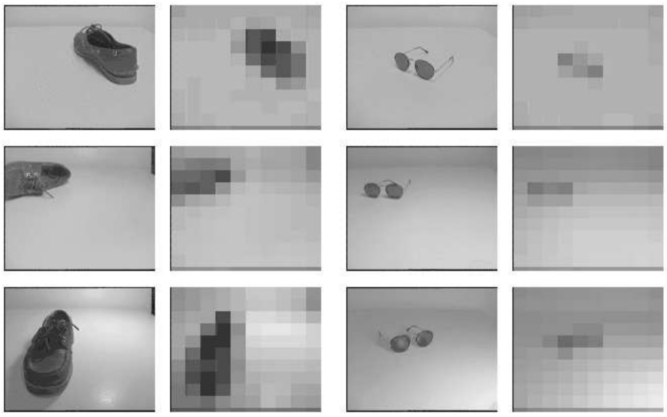

Figure 3: Images, along with the corresponding network inputs, of the objects shoe and glasses. These examples illustrate some of the invariances in the object recognition domain.

The objects were chosen so as to provide color and size cues helpful to their discrimination. The background of all images consisted of plain, white cardboard. Different images of the same object varied by the relative location and orientation of the object within the image. In 50% of all snapshots, the location of the light source was also changed, producing bright reflections at random locations in various cases. In some of the images the objects were back-lit, in which case they appeared to be black. Fig. 3 shows examples of two of the objects, the shoe and the glasses.

Images were encoded by a 300-dimensional vector, providing color, brightness and saturation information for a down-scaled image of size 10 by 10. Examples for the down-scaled images are shown in Figures 2 (right columns) and 3. Although each object appears to be easy to recognize from the original image, in many cases we found it difficult to visually classify objects from the subsampled images. However, subsampling was necessary to keep the networks to a reasonable size.

The set of target functions, F, was the set of functions that recognize objects, one for each object. For example, the indicator function for the bottle, f_{bottle}, was 1, if the image showed a bottle, and 0 otherwise. Since we only presented distinct objects, all sets of positive instances were disjoint. Consequently, F obeyed the invariance property. The set of hypotheses H was the set of all artificial neural networks with 300 input units, 6 hidden units, and 1 output unit, as such a network was employed to represent the target function.

The objective was to learn to recognize shoes, i.e., $f^* = f_{\text{shoe}}$. Five other objects, namely the bottle, the hat, the hammer, the can and the book, were used to construct the support sets Y. To avoid any overlap in the training set X and the support sets in Y, we exclusively used pictures of a seventh object, glasses, as counterexamples for f_{shoe}. Each of the five support sets in Y, X_{bottle}, X_{hat}, X_{hammer}, X_{can} and X_{book}, contained 100 images of the corresponding object (positive examples) and 100 randomly selected images of other objects (negative examples). When constructing training examples for the invariance network, we randomly selected a subset of 1,000 pairs of images, 800 of which were used for training and 200 for cross-validation. 50% of the final training and cross-validation examples were positive examples for the in-

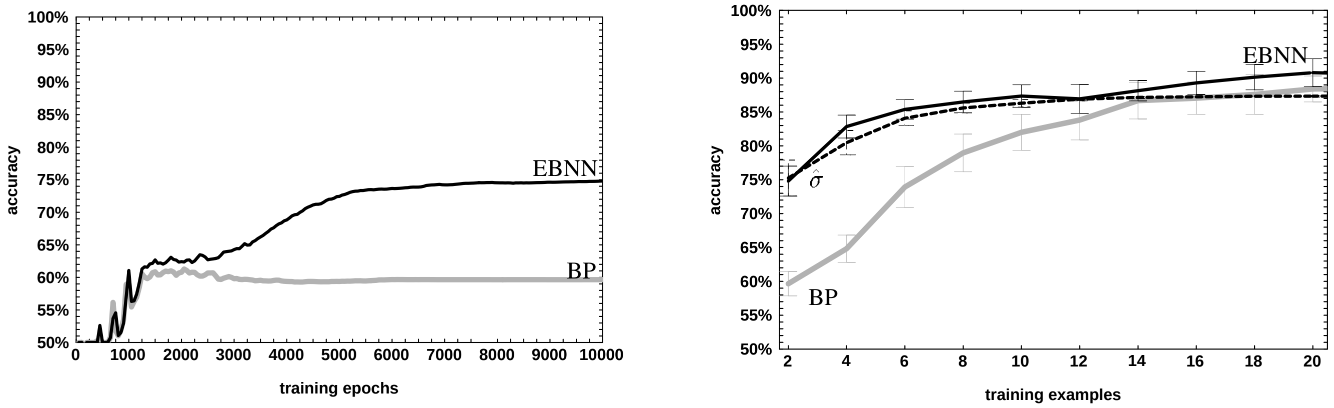

Figure 4: Generalization accuracy, with (solid black curve) and without (gray curve) the invariance network and EBNN, measured on an independent test set and averaged over 100 runs: (a) neural network training curves, one training example per class, and (b) generalization curves with 95% confidence intervals, as a function of the number of training examples.

variance network (i.e., both images showed the same object), and the other 50% were negative examples. The invariance network was trained using the Back-Propagation algorithm[2] After training, the invariance network managed to determine whether or not two objects belong to the same class with 79.5% generalization accuracy. It also exhibited 67.0% classification accuracy in the new task, the recognition of the shoe.

3.2 Learning to Recognize Shoes

Having trained the invariance network, we were now interested in training the classification network, f_{shoe}. The network employed in our experiments consisted of 300 input units, 6 hidden units, and 1 output unit—no effort was made to optimize the network topology. A total of 200 examples of images showing the shoe and the glasses were available for training and testing the shoe classification network. In our first experiment, we trained the classification network using only two of these: a randomly selected image of the shoe (positive example), and a randomly selected image of the glasses (negative example). Slopes were computed using the previously learned invariance network.[3]

Our experiments mainly addressed the following two questions, which are central to the lifelong learning framework and the invariance approach:

1. How important are the support sets, *i.e.*, to what extent does the invariance network improve the generalization accuracy when compared to standard supervised learning?
2. How effectively can EBNN overcome errors in the invariance network? How does EBNN compare to using the invariance network as a learned, generalized distance metric (*cf.* Eq. (4))?

Fig. 4a shows the average generalization curve as a function of training epochs with and without the invariance network. The curve shows the generalization accuracy of the classification network, each trained using one positive and one negative example. Without the invariance network and EBNN, the average generalization accuracy for Backpropagation is 59.7%. However, EBNN increases the accuracy to 74.8%. The invariance network alone, when used as generalized distance metric, classifies 75.2% of unseen images correctly. Notice the accuracy of random guessing would be 50.0%.

The difference between the performance with and without support sets, which is statistically significant to the 95% level, can be assessed in several ways. In terms of *residual error*, Backpropagation exhibits a misclassification rate that is 60.1% larger than that of EBNN. A second interpretation is to look at the *performance increase*, which is defined as the difference in classification accuracy after learning and before learning, assuming that the accuracy before learning is 50%. EBNN's performance increase is 24.8%, which is 2.6 times better than Backpropagation's 9.7%. The difference between EBNN and the invariance network are not statistically significant (at the 95% confidence level).

Notice that each of these results has been obtained by averaging 100 experiments. Examining a single experiment provides additional insight. For example, when the neural network is trained using the single image of the shoe and the single image of the glasses depicted in Fig. 2, plain Backpropagation classifies only 52.5% of the test images correctly. Here the generalization rate is particularly poor, since the location of the objects within the image differs, and Backpropagation mistakenly considers location the crucial feature for object recognition. EBNN produces a network that is much less sensitive to object location, resulting in a 85.5% generalization accuracy in this particular experiment.

These results illustrate that when training data for the target function is scarce, support sets can significantly boost generalization accuracy. They also illustrate that EBNN manages to make very effective use of the invariance knowledge captured in $\hat{\sigma}$. Notice that the results summarized above refer to the classification accuracy after 10,000 training epochs, using just one positive and one negative training example. As can be seen in Fig. 4a, Backpropagation suffers from some over-fitting, as the accuracy drops after a peak at about 2,050 training epochs. The average classification accuracy at this point in time is 61.3%. However, due to lack of data, it is im-

[2] The classification accuracy of the invariance network was significantly improved using a technique described in [Suddarth and Kergosien, 1990]. See [Thrun and Mitchell, 1994] for details.

[3] Since in our experiment the negative class, *i.e.*, the glasses, forms itself a disjoint class of images, those images can are also used to derive slopes (the slopes of $\hat{\sigma}$ were simply multiplied by -1). This effectively doubles the number of slopes considered in Eq. (5). The corresponding probabilities $1 - \hat{\sigma}(i, i_{\text{neg}})$ can also be incorporated into Eq. (4). See [Thrun and Mitchell, 1994] for details.

possible in this domain to use early stopping methods that rely on cross validation, and it is not clear that such methods would have improved the results for Backpropagation significantly.

Results for experiments with larger training set sizes are depicted in Fig. 4b. As the number of training examples increases, Backpropagation approaches the performance of EBNN. After presenting 10 randomly drawn training examples of each class, EBNN classifies 90.8% and Backpropagation classifies 88.4% of the testing data correctly. This matches our expectations, as the need for background knowledge decreases as the number of training examples increases. The invariance network alone using Eq. (4) (dashed curve) performs slightly worse than both of these methods. Its generalization accuracy is 87.3%, which is significantly worse than that of EBNN (at the 95% confidence level).

3.3 The Role of the Invariance Network

The improved classification rates of EBNN, which illustrate the successful transfer of knowledge from the support sets via the invariance network, raise the question of what exactly are the invariances represented in this network. What type information do the slopes convey?

A plausible (but only approximate) measure of the importance of a feature is the magnitude of its slopes. The larger the slopes, the larger the effect of small changes in the feature on the classification, hence the more relevant the feature. In order to empirically assess the importance of features, average slope magnitudes were computed for all input pixels, averaged over all 100 pairs of training instances. The largest average slope magnitude was found for color information: 0.11. In comparison, saturation slopes were, on average, only 0.063 (this is 57% of the average for color slopes), and brightness slopes only 0.056 (51%).

These numbers indicate that, according to the invariance network, color information was most important for classification. To verify this hypothesis, we repeated our experiments omitting some of the image information. More specifically, in one experiment color information was omitted from the images, in a second saturation, and in a third brightness. The results

	without inv. net	with inv. net
no color	52.4%	57.9%
no saturation	59.0%	72.9%
no brightness	58.7%	76.3%
full information	59.7%	74.8%

confirmed our belief that color information indeed dominates classification. It is clear that without color the generalization accuracy over the test set is poor, although EBNN still generalizes better. If saturation or brightness is omitted, however, the generalization rate is approximately equivalent to the results obtained for the full images reported above. However, learning required significantly more training epochs in the absence of brightness information (not shown here).

Fig. 5 shows average slope matrices for the target category (shoes) with respect to the three input feature classes, measuring color, brightness and saturation. Grey colors indicate that the average slope for an input pixel is zero. Bright and dark colors indicate strongly positive and strongly negative slopes, respectively. Notice that these slopes are averaged over all 100 explanations used for training.

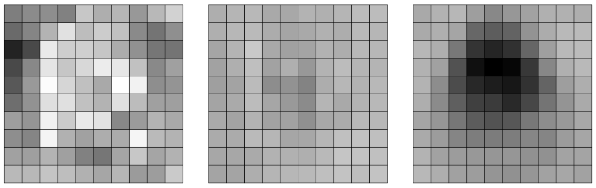

Figure 5: Slopes of the target concept (glasses) with respect to (a) color, (b) saturation, and (c) brightness. White (black) color represents positive (negative) values.

As is easily seen, average color slopes vary over the image, showing a slight positive tendency on average. Average saturation slopes are approximately zero. Brightness slopes, however, exhibit a strong negative tendency which is strongest in the center of the image. One possible explanation for the latter observation is the following: Both the shoe and the glasses are dark compared to the background. Shoes are, on average, larger than glasses, and hence fill more pixels. In addition, in the majority of images the object was somewhere near the center of the image, whereas the border pixels showed significantly more noise. Lack of brightness in the image center is therefore a good indicator for the presence of the shoe, as is clearly reflected in the brightness slopes derived from the invariance network. The less obvious results for color and saturation might be attributed to the fact that optimal classifiers are non-linear in color and saturation. In order to discriminate objects by color, for example, the network has to spot a specific interval in color space. Hence, the correct slopes can be either positive or negative depending in the particular color of a pixel, cancelling each other out in this plot.

As pointed out earlier, slopes provide first-order information, and invariances may well be hidden in higher-order derivatives. However, both the superior performance of EBNN and the clear correlation of slope magnitudes and generalization accuracy show that EBNN manages to extract useful invariance information in this domain, even if these invariances defy simple interpretation.

3.4 Using Support Sets as Hints

A related family of methods for the transfer of knowledge across learning tasks are proposed in [Suddarth and Kergosien, 1990], citePratt93aABBREV, [Caruana, 1993]. In a nutshell, these approaches develop improved internal representations by considering multiple functions in F (sequentially, or simultaneously). Following these ideas, we trained a single classification network providing the support data as "hints" for the development of more appropriate internal representations. This approach resulted in 62.1% (20 hidden units), or 59.8% (5 hidden units) generalization accuracy when only a single pair of training instances was used. These numbers can directly be compared to the experiments reported above. However, we observed significant overfitting when using this architecture. The peak generalization rate of 70.6% (20 hidden units), or 69.8% (5 hidden units), respectively, occurred after approximately 450 training epochs. This generalization accuracy is significantly higher than that of standard Backpropagation, though not as high as that of the invariance approach with EBNN.

4 Discussion

In the lifelong learning framework, the learner faces a collection of related learning tasks. The challenge of this framework is to transfer knowledge across tasks, in order to generalize better from fewer training examples of the target function itself.

This paper investigates a particular type of lifelong learning, in which binary classifiers are learned in a supervised manner. In the approach taken here, invariances are learned and transferred using the EBNN learning algorithm. The experimental results provide clear evidence of superior generalization in the object recognition domain, when invariances learned from related tasks are used to guide generalization when learning to recognize a new object. However, the the invariance approach relies on several critical assumptions:

1. Well-defined invariance functions rest on the assumption that F obeys the invariance property. Note even if the invariance property is not satisfied by F, the support sets can be used to train an invariance network. Even the object recognition domain presented above provides an example in which the invariance property may hold only approximately. This is because different objects may look alike in sufficiently coarse-grained, noisy images.
2. It is also assumed that functions in F possess certain invariances which can actually be learned by the invariance network. This fact does not follow from the invariance property. The exact invariances that will be learned depend crucially on the input representation and function approximator used for $\hat{\sigma}$.
3. We also assumed that the output space O of functions in F is binary. However, this assumption is not essential for the invariance approach. In principle, invariance functions may be defined for arbitrary, high-dimensional output spaces, given that a notion of difference between output vectors is available, as demonstrated in [Thrun and Mitchell, 1994].

In the experiments reported above, all three assumptions were at least approximately fulfilled. We conjecture that the real world offers a variety of tasks where learned invariances can boost generalization. Problems such as face recognition, cursive handwriting recognition, stock market prediction and speech recognition, possess non-trivial but important invariances. For example, consider the problem of learning to recognize faces of various individuals. Here certain aspects are important for successful recognition (e.g., the distance between the eyes), whereas others are less important (e.g., the direction in which the person is looking). After training on a number of individuals, we conjecture that an invariance network might grasp some of these invariances, reducing the difficulty of learning faces of new individuals.

The central question raised in this paper is whether learning can be made easier when the learner has already learned other related tasks. Will a system that is “trained” to learn generalize better than a novice learner? This paper provides encouraging results in an object recognition domain. However, most questions that arise in the context of lifelong learning still lack satisfactory, more general answers. We expect that future research in this direction will be important to going beyond the intrinsic bounds associated with learning single isolated functions.

Acknowledgment

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. Views and conclusions contained in this document are those of the authors 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

[Baxter, 1995] J. Baxter. The canonical metric for vector quantization. 1995.

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

[Hild and Waibel, 1993] 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, April 1993.

[Lando and Edelman, 1995] M. Lando and S. Edelman. Generalizing from a single view in face recognition. Technical report, Department of Applied Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot 76100, Israel, January 1995.

[Mitchell and Thrun, 1993] T.M. Mitchell and S. Thrun. Explanation-based neural network learning for robot control. In S. J. Hanson, J. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 287–294, San Mateo, CA, 1993. Morgan Kaufmann.

[Moore *et al.*, 1992] 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.

[Pratt, 1993] L.Y. Pratt. Discriminability-based transfer between neural networks. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 5*, San Mateo, CA, 1993. Morgan Kaufmann.

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

[Simard *et al.*, 1992] P. Simard, B. Victorri, Y. LeCun, and J. Denker. Tangent prop – a formalism for specifying selected invariances in an adaptive network. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 895–903, San Mateo, CA, 1992. Morgan Kaufmann.

[Sinha, 1994] P. Sinha. Object recognition via image-invariants. Presented at the 1994 ARVO Conference, Florida, 1994.

[Suddarth and Kergosien, 1990] S.C. Suddarth and Y.L. Kergosien. Rule-injection hints as a means of improving network performance and learning time. In *Proceedings of the EURASIP Workshop on Neural Networks*, Sesimbra, Portugal, 1990.

[Sutton, 1992] 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. AAAI, AAAI Press/The MIT Press, 1992.

[Thrun and Mitchell, 1994] S. Thrun and T.M. Mitchell. Learning one more thing. Technical Report CMU-CS-94-184, Carnegie Mellon University, Pittsburgh, PA 15213, 1994.

[Thrun and Mitchell, to appear] S. Thrun and T.M. Mitchell. Lifelong robot learning. *Robotics and Autonomous Systems*, to appear. Also appeared as Technical Report IAI-TR-93-7, University of Bonn, Dept. of Computer Science III, 1993.

[Utgoff, 1986] P.E. Utgoff. Shift of bias for inductive concept learning. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach, Volume II*. Morgan Kaufmann, 1986.