Latent Learning in Deep Neural Nets

Steven Gutstein, Olac Fuentes and Eric Freudenthal

***Abstract*— Psychologists define *latent learning* as learning that occurs without task-specific reinforcement and is not demonstrated until needed. Since this knowledge is acquired while mastering some other task(s), it is a form of transfer learning.**

We utilize latent learning to enable a deep neural net to distinguish among a set of handwritten numerals. The accuracies obtained are compared to those achievable with a simplistic 'group-mean' classification technique, which is explained later in this paper. The deep neural net architecture used was a LeNet 5 [3] convolutional neural net with only minor differences in the output layer.

I. Introduction

Transfer learning involves using knowledge acquired from one set of tasks in order to help master one or more related tasks. The reuse of previously obtained knowledge enables new tasks to be learned with less effort, as measured mainly by training set size and time required for learning. What has previously been overlooked is that for sufficiently related sets of tasks, mastery of one set may immediately grant a degree of mastery for the other. Learning, which occurs without reinforcement and is not demonstrated until it is needed, is referred to as *latent learning*. It was first demonstrated in 1930's in a set of experiments with mice, performed by Tolman and Honzik [9].

Training of a neural net can be extremely time consuming. During standard training, the internal parameters of a neural net are continually adjusted until one manages to get the net to behave in a desired manner. With latent learning, although the net is trained for an initial set of tasks, it is not trained for the subsequent set. The behavior of the net is unchanged when one tries to apply it to the new task(s). So, the net retains full knowledge of the first set of tasks.

The 'training' only occurs with respect to the classifier of the net's output. The classifier must learn how to properly interpret the output of the net. As will be discussed, this is fairly trivial to do. For lack of a better term and a desire to avoid creating new terminology unless necessary, the set of labeled samples used to help learn the net's behavior will be referred to as the 'training' set. However, it must be stressed that when relying solely upon latent learning, this dataset is not used to modify any of the neural net's internal parameters or to alter its behavior. It is merely a training set with which the net is not trained. One can also think of it as a 'sample set'.

The results we achieved with latent learning are far better than random guessing would be expected to achieve. Additionally, they are comparable with results achieved by others who used small training sets in the traditional sense of 'training', along with another approach to transfer learning [1].

Steven Gutstein, Olac Fuentes and Eric Freudenthal are with the Department of Computer Science, University of Texas at El Paso, 500 West University Av., El Paso, TX 79968 (email: s.m.gutstein@gmail.com).

There is no conflict with using latent learning and then actually training on the samples in the training set. Earlier work [2] has demonstrated that by only allowing the top layer of a deep net to train, in addition to using latent learning, it is possible to gain significant increases in accuracy. Furthermore, even though only an exceedingly small portion of the net's free parameters may be affected by such training, they suffice to obtain almost all of the benefit that could be obtained if the entire net were to train. However, there will be some degradation of the net's performance on the original set of tasks.

The purpose of this paper is two-fold. First, we investigate the degree of improvement that a pure latent learning approach can achieve, merely by increasing the number of output nodes in the neural net. Second, we make a comparison between the latent learning results and a group mean classification technique.

The group-mean classification starts with a training set with several examples from each of k groups. For each group, it calculates an 'average' of all the labeled samples from that group. There are some slight adjustments made to these averages, which will be explained later. Unlabeled samples are then classified based upon to which of the k group averages they are most similar.

The point of this comparison is to demonstrate the efficacy of latent learning, particularly for very small training sets, such as 1 sample/class. It is also meant to emphasize the fact that true transfer learning does occur with the learning procedure we use.

A. Inherent Bias and Latent Learning

The concept of inherent bias is fairly straightforward. Prior to any training, when a net has only been randomly initialized, any given input will cause some of the output nodes to fire and others not to fire. This set of firing/non-firing nodes will be referred to as the net's output. For any given class of input, the probability that a given output node will fire is **not** 50%. Each output class will tend to have a most probable output pattern. These most probable output patterns reflect the *inherent bias* of the net. When the various input classes have inherent responses that are sufficiently different and those responses have probability distributions that are sufficiently narrow, they can be used to distinguish among those input classes. This is prior to training of any sort with only a randomly initialized neural net. The first use of inherent bias to distinguish among a set of input classes is described in Gutstein et al.[2].

One way to determine a net's inherent bias is to use a training set with labeled samples of each class in order to calculate the average output for each node. Then, assuming the nodes have a transfer function that associates +/-1 with a node firing/not firing, let the inherent response for each individual node to a given class be $+1$, if its average response to that class is ≥ 0, and -1, otherwise. There are, obviously, other, more precise ways to do this, but for these initial experiments, this method more than suffices.

More sophisticated approaches would include only giving a node a preferred firing value if the magnitude of its average response were above a certain threshold, or attaching varying importance to the estimated value of the output node. This importance could be determined either by a value associated with the magnitude of the average response, or with how well the firing/not firing of that node correlated with the sets of classes for which it is supposed to fire/not fire. These possibilities will be explored in future papers.

A drawback to our approach is that some nodes will be assigned incorrect values for their inherent response and other nodes, whose response is not correlated with the given class, will still be assigned a response. As the number of output nodes increases, the question must be asked how quickly the number of these maladaptive nodes grows. Offsetting the growth of these maladaptive nodes will be the growth of the nodes with correctly estimated and useful inherent responses. Furthermore, one must consider whether the distance between the inherent responses grows more quickly than the distribution of actual responses generated by actual data around those responses (i.e. do the inter-class distances grow more quickly than the intra-class distances ?). For these reasons, we wanted to investigate the effect of increasing the number of output nodes in the neural net on latent learning.

B. Convolutional Neural Nets

The neural net architecture we used is a LeNet-5 style convolutional neural net [3] as shown in Figure 1.

The advantage of this architecture is that it imposes a bias that limits the net to hypotheses that are particularly useful for recognition of handwritten characters, which is the problem we are using to investigate latent learning. The allowed hypotheses display:

1) Translational Invariance
2) Moderate Insensitivity to Rotations
3) Moderate Insensitivity to Geometric Distortions
4) Dependence upon Small-scale, Local Correlations

One of the architectural constraints that enforces these conditions is restricting the inputs for each node to small, locally connected regions. Rather than connecting each node to all the nodes in the layer beneath it, a node is only given connections to a small, adjacent grouping of nodes in the layer beneath it. The effect of this is to bias each node towards small-scale local correlations in that layer for use as significant features.

Furthermore, the layers of the net are divided into several feature maps (See Figure 1). Nodes in the same feature map have the same set of input weights. This is known as 'weight sharing'. Nodes with shared weights all react to the same set of features. Additionally, the spatial relationship between two nodes in a given feature map will be replicated with their receptive fields. It should be noted that a given node may have receptive fields in several feature maps in the layer immediately below it. When this happens, all nodes in the same feature map as the given node will have receptive fields with identical spatial relationships in the same set of feature maps. Therefore, each such map will detect and locate a given feature or combination of features from the previous layer no matter where that given feature(s) appears in the maps of the previous layer. This provides the translational invariance.

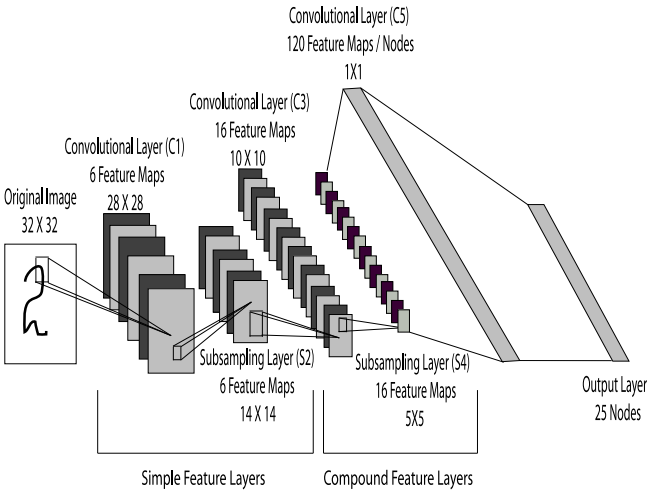

Fig. 1. Architecture of our net, which is a slightly modified version of LeNet5 as shown in [3]. It should be noted that the feature maps in the C5 & Output layers are 1 node $\times$ 1 node. So, they could with equal accuracy be considered as traditional nodes in a non-weight sharing feed-forward neural net.

There are two standard types of layers found in a CNN: convolutional and sub-sampling layers. In the convolutional layers (C-layers) of the net, adjacent nodes will have overlapping receptive fields with the same relative spatial positions. In the other type of layer, a sub-sampling layer (S-layer), each feature map is connected to exactly one feature map of the prior layer. Additionally, the receptive fields of nodes in a sub-sampling map do not overlap. So, while the C-layers perform what may be thought of as actual feature detection, the S-layers perform a local averaging. It is this local averaging that provides the moderate insensitivity to rotations and geometric distortions. Finally, the layered structure of the CNN means that the first C-layer contains feature maps that detect simple features, later C-layers will tend to detect compound features. A more detailed description of CNN's may be found in LeCun et al. [3]

II. Main Results

A. Creating the source net for latent learning

All our experiments started with a net that had been trained to recognize the 25 characters A-F,H,I,P,R-W,Y,a,b,d,e,g,h,q,r

and t.

The dataset from which we obtained samples of these characters was the NIST Special Database 19, which contains 62 classes of handwritten characters corresponding to '0'-'9', 'A'-'Z' and 'a'-'z'. Our choice of which subset of characters to train upon was governed mainly by the number of samples of each character and the desire to avoid training on either the letter 'O' or 'o', since we felt both were too similar to the number '0' for our purposes.

The nets were initially trained to respond to each of the 25 letters with a unique pattern of the output nodes either firing or not firing. These patterns were thought of as points in a 25 dimensional space and will be referred to as 'target' points. The specific target point for each character class was chosen randomly. Each node had a 50% chance of being set to either firing or not firing for the target point of a particular class. Input images were classified based upon to which target point their output was closest in a Euclidean sense. Training consisted of a single training epoch over a set of images with 775 samples per letter. Empirically, this gave the best results for latent learning.

Once a net was trained on the first set of 25 letters, it was introduced to samples from the classes '0'-'9'. To estimate the inherent bias of the net with respect to this new set, we simply calculated the average output of the net for each class. The desired output (i.e. target point) for each class was determined by calculating the sign function (i.e. $sgn(x)$) of each of the average output points. These were then used to classify characters belonging to the classes '0'-'9'. This process may be thought of as 'learning' the net.

For example, if we had been using only 3 output nodes and the average output of the *n* samples in our training set of the character '7' was [0.6, -0.8, 0.9], the target point for '7' would have been [1.0,-1.0,1.0]. We felt this was a reasonable estimation of the the net's inherent bias for character '7'.

Clearly, this procedure does not effect the net's response to input in any way. This is the basis of our claim that the net has not been trained for the new set of tasks, even though a 'training' set is used to estimate the net's inherent bias.

B. Group-Mean Classification Comparison

Our implementation of latent learning might be considered to be no more than a needlessly complex clustering technique. We determined the inherent bias of the net by looking at its average output for each class and then assigning a value of $+1/-1$ to each node, depending upon whether or not the average output was positive or negative.

A comparable technique, which would avoid the complication of the neural net would be to perform the same type of averaging with our raw data. These averages could then be used for a group-mean classification, where any new sample would be classified according to which 'group-mean' it was closest, in a Euclidean sense. Since our images were binary images, it seemed reasonable to adjust our averages to reflect this. So, if the average for a certain pixel over a set of images belonging to the same class was ≥ 127.5 the value of the average was set to 255, otherwise, the average was set to 0. This made sense because our images were binary images (i.e. pixel values were either 0 or 255). It also made our clustering method more similar to the way the latent responses of the net were determined.

If we are truly benefiting from latent learning, and not just learning from information present in the raw, task-specific data, then we would expect our latent learning technique to achieve markedly better accuracies than our group-mean classification technique. To further demonstrate the existence of latent learning, we will examine the accuracies realized by nets that have not experienced any training and rely solely upon their inherent biases in order to classify inputs.

C. Effects of Increasing the Number of Output Nodes

In order to determine the effect of increasing the number of output nodes on latent learning, we created sets of 5 LeNet-5 style neural nets with 25, 100, 400, 1600 and 6400 output nodes, using the method described earlier. Then, we also prepared 5 sets of datasets from the set of numeric characters '0' - '9', consisting of training set sizes with 1, 5, 10, 20, 40, 80 & 160 samples per class (i.e. 10, 50, 100, 200, 400, 800 & 1600 samples) and testing sets with 100 samples per class (i.e. 1,000 samples). Each training set had its own testing set.

After determining the inherent bias for each training set, as earlier described, the accuracy obtained by each net on each corresponding testing set was measured. In Figure 2, the average of the accuracies for each of the 5 nets and each of the 5 datasets are shown for each architecture. Each data point is an average over 25 values.

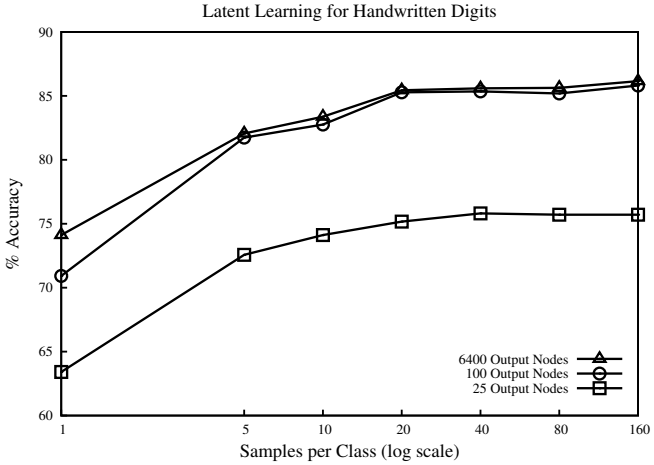

Fig. 2. Average accuracies obtained for latent learning for various sizes of training sets and number of output nodes. The size of the training sets are given in log-scale on the x-axis in terms of samples per class. Since we were using the set of numeric characters '0' - '9', there are 10 classes.

Figure 2 shows that it was possible to obtain accuracies ranging from just under 65% to just under 75% even when the net is exposed to only a single sample from each of the 10 classes it is expected to discriminate among. With larger training sets, it is possible to achieve accuracies in excess of 85%, which we reiterate is done without training the net in any way for this particular problem. The training set only

provides samples to help characterize the existing behavior of the net, not to change it. Adding additional output nodes to the neural net appears to be purely beneficial. This holds even when the number of output nodes begins to equal or exceed the original number of input values.

The images we used were 32×32 pixels, giving them only 1,024 values. Although experiments were performed with nets possessing 25, 100, 400, 1600 and 6400 output nodes, results are only shown for 25, 100 and 6400 output nodes. This is because the accuracy change from 100 output nodes to 6400 output nodes is very slight.

The numbers of output nodes examined have the following history behind their choice. Originally, 25 output nodes were used, because the latent learning was based on first training with 25 classes of letters. Since target points were assigned by randomly assigning each output node a target value of $+1/-1$, the average Euclidean distance between two such target points grows as the square root of the number of output nodes. So, in order to double the expected distance between any two randomly created targets, we had to quadruple the number of output nodes. In retrospect, perhaps, a more important quantity is the ratio of the number of output nodes to the number of input nodes. These ratios are 0.02, 0.10, 0.39, 1.56 and 6.25.

The values presented in Figure 2 are average accuracies. In order to gain an appreciation of how much one might expect accuracies to vary, results are given showing the maximum, average and minimum results obtained by the net with 25 output nodes and the net with 6400 output nodes. These results are shown in Figures 3 and 4.

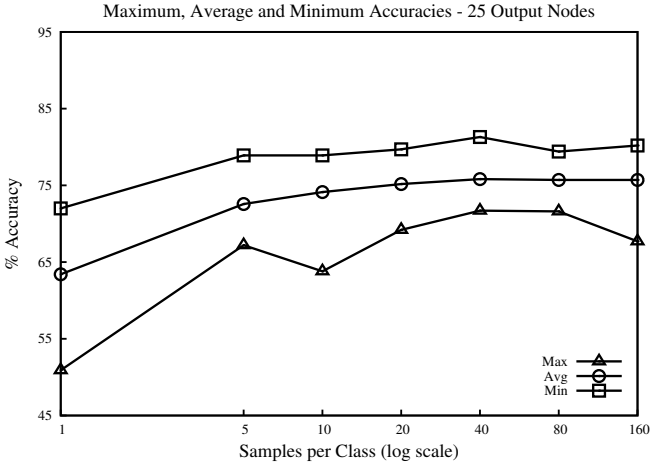

Fig. 3. Maximum, average and minimum accuracies obtained for latent learning for various training set sizes and for a net with 25 output nodes. The size of the training sets are given in log-scale on the x-axis in terms of samples per class. Since we were using the set of numeric characters '0' - '9', there are 10 classes.

Earlier results from Gutstein et al.[2] suggest that if we were to allow only the output nodes to train in addition to using latent learning, then significant increases in accuracy could be anticipated. Furthermore, limiting training to only the output nodes means that only a fraction of the net's parameters are trained. In the case of the 25 output node net, only about 5% of the net's nodes are trained.

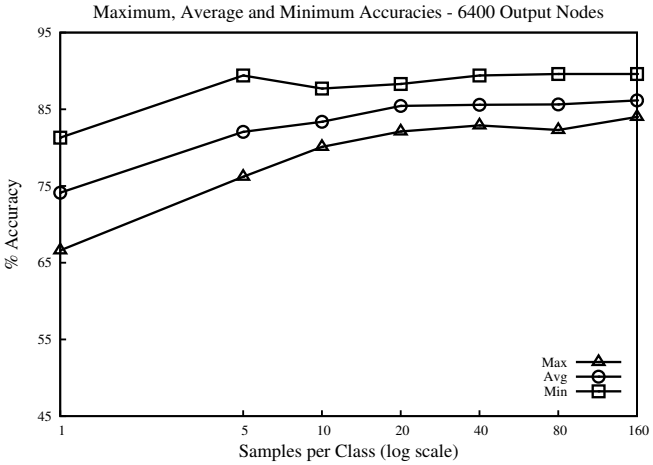

Fig. 4. Maximum, average and minimum accuracies obtained for latent learning for various training set sizes and for a net with 6400 output nodes. The size of the training sets are given in log-scale on the x-axis in terms of samples per class. Since we were using the set of numeric characters '0' - '9', there are 10 classes.

D. Comparison with Group-Mean Classification

Given the way we determine the inherent bias for latent learning, it is reasonable to ask whether or not all we have demonstrated is a complex clustering technique. Since, we essentially take an average of the neural net's outputs for each class, round the average values of each output node to +/-1, then classify each subsequent input by determining to which class average it is closest. So, to verify that there is something more going on, the group-mean technique described earlier was used with each of our datasets. A comparison between the results thus obtained and those obtained from some of our previous experiments is shown in Figure 5.

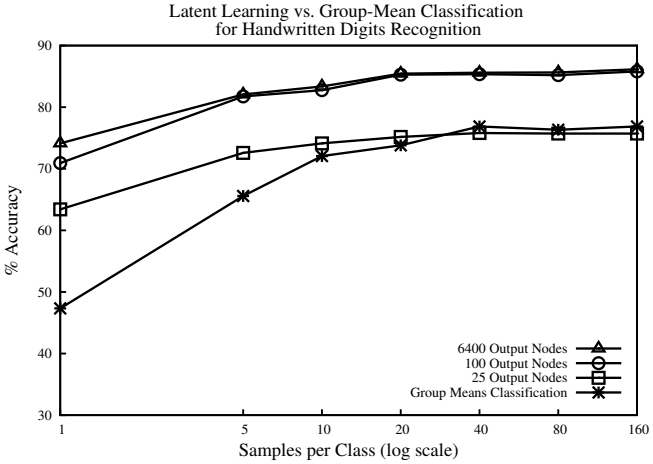

Fig. 5. Average accuracies obtained for latent learning vs. group-mean classification for various sizes of training sets and number of output nodes. The size of the training sets are given in log-scale on the x-axis in terms of samples per class. Since we were using the set of numeric characters '0' - '9', there are 10 classes.

When examining Figure 5, it is important to realize that the nets with 25 & 100 output nodes are compressing the image

by factors of about 0.02 and 0.10 respectively. It is interesting to note that the asymptotic value of accuracy achieved by latent learning for the recognition of handwritten characters essentially occurs when the ratio of output nodes to input nodes is 0.10.

As the training sets grow larger, the performance of the group-mean classification method improves in both absolute and relative terms. Yet, it only manages to slightly exceed the accuracy achieved by the net with 25 output nodes when the number of samples per class in the training set becomes relatively large.

The advantage in accuracy is due to latent learning and not just the use of the net's inherent bias. This is seen by comparing the accuracies realized by nets that use only their inherent bias to perform this recognition task with those that also employ latent learning. Using Figure 5 with Figure 6 for such a comparison, one can see that latent learning not only occurs, but that it is a significant source of learning.

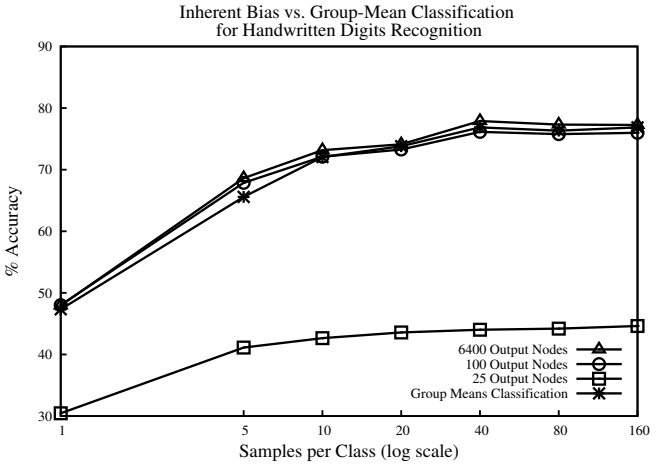

Fig. 6. Average accuracies obtained with inherent bias vs. group-mean classification for various sizes of training sets and number of output nodes. The size of the training sets are given in log-scale on the x-axis in terms of samples per class. Since we were using the set of numeric characters '0' - '9', there are 10 classes. It should be noted that the accuracy realized by the group-mean classification method is virtually indistinguishable from those achieved by the 100 & 6400 output node nets, that relied solely upon their inherent bias.

III. Conclusions

Although latent learning has been well known to psychologists for approximately 80 years, it has not been a common technique for machine learning. However, the benefits it promises in terms of learning from small datasets and enabling lifelong learning make it a promising area of research. In this paper, we have demonstrated that increasing the number of output nodes in neural net seems to only increase the efficacy of latent learning, up to an asymptotic limit. We have also reaffirmed that there is actual transfer learning occurring. This occurs even though the parameters of the net and behavior of the net are in no way adjusted to solve the problem for which information is being transferred. The only adjustment made is learning to interpret the net's responses when it is exposed to the new problem.

Two issues that were not addressed in this paper are task relatedness and consolidation of sequentially acquired knowledge. Many researchers studying transfer learning put a great deal of effort into deciding whether the sets of tasks being considered are 'related' enough for knowledge transfer to be an effective technique [4], [5], [7] (e.g. knowledge of Spanish is extremely helpful when learning Italian, but is progressively less helpful when learning Chinese, American Sign Language or Figure Skating). Because the tasks we were considering were about as related as possible, while still being different, this was not an issue for us. If latent learning is to develop, methods for determining to what degree a latent response can be relied upon will have to be developed.

The other issue, which we avoided, is task consolidation. We have presented a technique that enables a neural net to use its ability to perform one set of tasks to enable it to more readily perform a second set of tasks. However, our net can only be used to perform either the first set of tasks (i.e. discriminating among the 25 chosen letters) or the second set of tasks (i.e. discriminating among the 10 numbers). It cannot currently be used to reliably perform both sets of tasks at the same time (i.e. discriminating among the 25 chosen letters and the 10 numbers). We hope to overcome this problem through finding ways to ensure better output codes, that will lend themselves to latent learning, and using a task-rehearsal method [6], [8] or some variant of it.

Additionally, there is still work to be done in finding more precise methods of determining a net's inherent bias and in determining how to create, or at least predict which nets that will have most successfully experienced latent learning.

Acknowledgment

This work was supported in part by the National Science Foundation under grant no. CNS0454189 and by Texas Instruments.

References

[1] James Bergstra and Yoshua Bengio, "Decorrelated Features for Pre-training Complex Cell-like Networks," *Advances in Neural Information Processing Systems 22 (NIPS'09)*, 2009.

[2] Steven Gutstein and Olac Fuentes and Eric Freudenthal, "Training to a Neural Net's Inherent Bias," *Proc. Proceedings of the Twenty-Second International Florida Artificial Intelligence Research Society Conference*, Sanibel Island, Florida, May, 2009.

[3] Yann LeCun and Leon Bottou and Yoshua Bengio and Patrick Haffner, "Gradient Learning Applied to Document Recognition", *Proc. Proceedings of the IEEE*, Nov. 1998, pp.2278–2374.

[4] Shai Ben-David and Reba Schuller, "Exploring Task Relatedness for Multiple Task Learning", *Proc. Conference on Learning Theory (COLT) 2003*, 2003, pp. 567–580.

[5] Daniel L. Silver and Robert E. Mercer, "Selective functional transfer: Inductive bias from related tasks", *Proc. Proceedings of the International Conference on Artificial Intelligence and Soft Computing*, 2001, pp. 182–189.

[6] Daniel L. Silver and Robert E. Mercer, "The task rehearsal method of life-long learning: Overcoming impoverished data", *Proc. 15th Conference of the Canadian Society for Computational Studies of Intelligence (AI'2002)*, 2002, pp. 90–101.

[7] Daniel L. Silver and Richard Alisch, "A measure of relatedness for selecting consolidated task knowledge", *Proc. Proceedings of the International Conference on Artificial Intelligence and Soft Computing Proceedings of the 18th Florida Artificial Intelligence Conference (FLAIRS05)*, 2005, pp. 399–404.

[8] Daniel L. Silver and Robert E. Mercer, "Sequential Inductive Transfer for Coronary Artery Disease Diagnosis", *Proc. International Joint Conference on Neural Networks*, Orlando, Florida, Aug. 2007.
[9] E. C. Tolman and C. H. Honzik, *'Insight' in rats* California: University of California Publications in Psychology, 1930.