Machine Life-Long Learning with *cs*MTL Networks

Daniel L. Silver and **Ryan Poirier**
Jodrey School of Computer Science
Acadia University, Wolfville, NS, Canada B4P 2R6
http://plato.acadiau.ca/courses/comp/dsilver/ danny.silver@acadiau.ca

Multiple task learning (MTL) neural networks are one of the better documented methods of inductive transfer of task knowledge (Caruana 1997). An MTL network is a feed-forward multi-layer network with an output node for each task being learned. The standard back-propagation of error learning algorithm is used to train all tasks in parallel. The sharing of internal representation in the hidden nodes is the method by which inductive bias occurs between related tasks within an MTL network (Baxter 1996).

Previously, (Silver & Mercer 2002; Silver & Poirier 2004) have investigated the use of MTL networks as a basis for developing machine lifelong learning (ML3) systems and have found them to have several limitations caused by the multiple outputs. In response to these problems, this article introduces *context-sensitive* MTL, or *cs*MTL, and describes a ML3 system that uses these networks to perform long-term consolidation of task knowledge and the inductive transfer of prior knowledge during new learning.

*cs*MTL

Like MTL, a *cs*MTL network has a feed-forward architecture of input, hidden and output nodes that uses the back-propagation of error training algorithm. However, *cs*MTL requires only one output node for learning multiple concept tasks and the input layer is divided into two parts: $\mathbf{x}$, a set of *primary* input variables for the tasks and $\mathbf{c}$, a set of inputs that provide the network with the *context* of each training example. The context inputs can be a set of task identifiers that associate each training example to a particular task. Alternatively, $\mathbf{c}$ can offer more specific environmental information (such as location and light level) and in this way index over a continuous domain of tasks.

With *cs*MTL the focus shifts from learning a task, $f(\mathbf{x})$ of a domain to learning a domain of tasks, $f'(\mathbf{c}, \mathbf{x})$. The entire representation of the network is used to develop hypotheses for all tasks, f'. This is a more continuous sense of domain knowledge where the objective is to learn internal representations that are shared by all tasks but index by a combination of the primary and context inputs. During learning, $\mathbf{c}$ selects an inductive bias over the hypothesis space of the network, H_{csMTL}, relative to the secondary tasks being learned in the network. Once f' is learned, if $\mathbf{x}$ is held constant, then $\mathbf{c}$ indexes over H_{csMTL}. If $\mathbf{c}$ is a vector of real-valued inputs and from the environment, it provides a grounded sense of task relatedness. If $\mathbf{c}$ is a set of task identifiers, it differentiates between otherwise conflicting examples and selects internal representation used by related tasks.

A Machine Lifelong Learning System

Figure 1 shows the proposed *cs*MTL ML3 system. It has two components; a temporary *short-term learning network* and a permanent *long-term consolidation cs*MTL *network*. The long-term *cs*MTL network is the location where domain knowledge is retained over the lifetime of the learning system. The weights of this network are updated only after a new task has been trained to an acceptable level of accuracy in the short-term learning network. The short-term network can be considered a temporary extension of the long-term network that adds representation (several hidden nodes and an output node) that is needed to learn the new task. At the start of short-term learning the weights associated with these temporary nodes are initialized to small random weights while the weights of the long-term network are frozen. This allows representational knowledge to be rapidly transferred from related tasks in the long-term network without fear of losing prior task accuracies.

Once the new task has been learned, the temporary short-term network is used to consolidate knowledge of the task into the permanent long-term *cs*MTL network. This is accomplished by using a form of functional transfer called *task rehearsal* (Silver & Mercer 2002). The method uses the short-term network to generate *virtual examples* for the new tasks so as to slowly integrate (via back-propagation) the task's knowledge into the long-term network. Additionally, virtual examples for the prior tasks are used during consolidation to maintain the existing knowledge of the long-term network. Note that it is the functional knowledge of the prior tasks that must be retained and not their representation; the internal representation of the long-term network will necessarily be updated as the new task in integrated.

The algorithm for short-term learning is: (1) Fix the representation of the long-term *cs*MTL network; (2) Initialize the hidden and output node connection weights, that are unique to the short-term network, to small random values; (3) Train and test the short-term network using available data; and

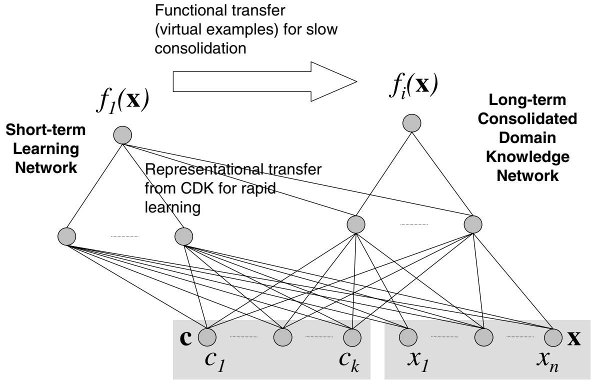

Figure 1: Proposed ML3 system base on *cs*MTL networks.

(4) If generalization accuracy is sufficient, consolidate into long-term network.

The algorithm for long-term consolidation is: (1) Generate virtual examples for new and prior tasks using the representations of the short-term and long-term networks; (2) Unfix the representation of the long-term network; and (3) Train the long-term network using the virtual examples.

Benefits and Limitations

The *cs*MTL ML3 system provides a number of benefits in terms long-term retention of learned knowledge. The approach provides effective retention of domain knowledge in the long-term *cs*MTL network via task rehearsal (Robins 1995; Silver & Mercer 2002). Retention is efficient in that related tasks share internal representation without the disadvantage of having separate outputs. The system is capable of increasing long-term task accuracy through practice without incurring redundant representation. The long-term *cs*MTL network can represent a fluid domain of tasks where subtle differences between tasks can be represent by small changes in the context inputs. We conjecture that *cs*MTL does not require an explicit method of indexing into domain knowledge for related tasks. Indexing occurs as the connection weights between the long-term network and the temporary short-term network are trained.

The *cs*MTL ML3 system also has benefits in terms of short-term learning. It employs rapid representational transfer from the long-term consolidation network. If the current task has been previously learned and retained, then the few weights between the long-term and the short-term networks will train quickly to produce the desired hypothesis. If the new task is different but related to a prior task, the long-term to short-term network weights will select the most appropriate features of long-term network and the supplemental hidden nodes of the short-term network will play only a partial role in the hypothesis. If the new task is unrelated to any prior learning, the supplemental representation of the short-term network will play the major role in the new hypothesis.

Experimental Results

An experiment was conducted to test the *cs*MTL ML3 system. The experiment compares the *cs*MTL method of inductive transfer in terms of efficiency (training time to develop a hypothesis) and effectiveness (test accuracy of a hypothesis) to STL (where there is no transfer), MTL (all tasks) and ηMTL (MTL with only the most related tasks). A domain of six synthetic non-linearly separable classification tasks were used where a positive example of each task is defined by a logical combination of 4 of the 10 real-valued inputs. Task $f_1 = (x_1 > 0.5 \vee x_2 > 0.5) \wedge (x_3 > 0.5 \vee x_4 > 0.5)$ with 20 training examples is the target task. Inductive transfer is from long-term MTL or *cs*MTL networks containing accurate hypotheses of the 5 other tasks.

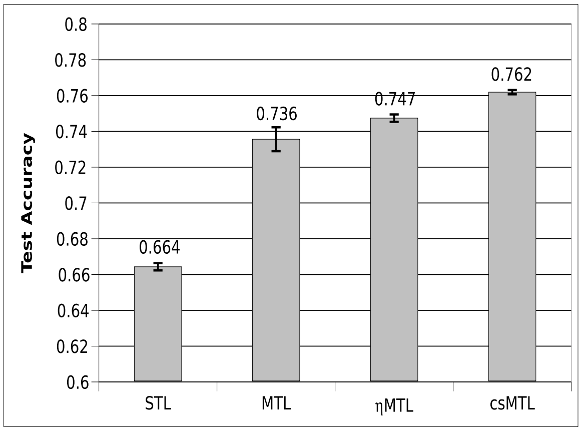

Figure 2: Experimental results: Shown is the mean test set accuracy (and 95% confidence intervals) for f_1 hypotheses .

Figure 2 shows the results for the four methods averaged over 30 trials. The graph indicates that the *cs*MTL ML3 system is able to selectively transfer knowledge from the long-term *cs*MTL network to a new task without the aid of a measure of task relatedness. Similarly, a comparison of the efficiency of model development by the four methods shows that STL is the most efficient, yet generates the least accurate hypotheses. MTL and ηMTL produce more effective hypotheses but are less efficient because they require parallel learning. *cs*MTL which requires only the original 20 training examples takes a little over twice as long to train as STL but it produces the best hypotheses.

References

Baxter, J. 1996. Learning model bias. In Touretzky, D. S.; Mozer, M. C.; and Hasselmo, M. E., eds., *Advances in Neural Information Processing Systems*, volume 8, 169–175. The MIT Press.

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

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

Silver, D. L., and Mercer, R. E. 2002. The task rehearsal method of life-long learning: Overcoming impoverished data. *Advances in Artificial Intelligence, 15th Conference of the Canadian Society for Computational Studies of Intelligence (AI'2002)* 90–101.

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