Abstract. We present context-sensitive Multiple Task Learning, or csMTL as a method of inductive transfer. It uses additional contextual inputs along with other input features when learning multiple tasks so as to improve predictive performance. We modified the WEKA machine learning suite to accept csMTL encoding multiple tasks examples and performed tests on three domains. The results are not as supportive as we expected, however they still demonstrate that inductive transfer with csMTL is beneficial.

1 Introduction

Inductive transfer learning involves the use of prior knowledge of one or more related secondary tasks while learning a primary task [5], so as to develop a more effective model that has greater generalization accuracy than developed with primary task only. One of the more common methods of inductive transfer is Multiple Task Learning (MTL) [1]. csMTL is a form of MTL that encodes data from multiple tasks using context inputs rather than multiple outputs [4]. This paper documents our testing of the well known WEKA machine learning suite [7] modified to allow its multi-layer perceptron algorithm to accept csMTL encoded data. Our objective is to increase the availability of inductive transfer systems to students, researchers and practitioners.

An MTL neural network is a feed-forward multi-layer network with a separate output for each task that is to be learned and one or more hidden layers of nodes that are common to all tasks [1]. The standard back-propagation of error learning algorithm is used to train all tasks in parallel. The sharing of internal representation is the method by which inductive bias occurs within an MTL network [1]. Previously, we have investigated the use of MTL networks and have found them to have several limitations related to the multiple outputs of the network [4]. Consequently, we have considered alternative methods.

In [4] we introduced context-sensitive MTL (csMTL), as a method of encoding examples of multiple tasks such that standard single task learning (STL) algorithms can, potentially, perform inductive transfer. csMTL
encoded examples have only one output but have additional inputs that indicate the example context, such as the task to which it is associated. A csMTL neural network is a single output back-propagation network that can accept csMTL encoded examples. The network may have one or more layers of hidden nodes. The input layer is composed of two parts: a set of primary input variables for the tasks and a set of inputs that provide the network with the context of each example.

csMTL overcomes the limitations of standard MTL for construction of a machine lifelong learning system [4]. First, csMTL eliminates redundant outputs for the same task making it easier to accumulate knowledge of a task through practice. Secondly, csMTL examples are associated directly with a task via the context inputs, c. Finally, we conjecture that relatedness between tasks can be measured by the similarity of the context c, if c is environmentally grounded [4].

2 WEKA and the csMTL MLP Algorithm

WEKA is an open source project of the University of Waikoto [7]. It has been widely used in colleges and universities as well as by data mining practitioners on many real-world domains [3]. The WEKA multi-layer perceptron (MLP) was implemented by Malcolm Ware in 2000 [6]. Its use has been documented in a number of research publications [2]. Our challenge is to implement a new version of WEKA MLP such that it is capable of working with csMTL encoded data.

Three major extensions to the existing WEKA MLP were implemented in a new class, MultilayerPerceptronCS (MLP_CS). The first is the implementation of the -T <filename> parameter which allows the specification of a data source file, from which a secondary task training set is drawn. This allows the use of csMTL encoded data. The other two extensions were implemented for convenience of testing. The second is the implementation of the -X <filename> parameter which allows the specification of a data source file, from which a validation set may be drawn. This allows greater flexibility than the existing validation option, which selects a percentage of the training set to act as a validation set. The third is the implementation of a display in the neural network GUI that indicates the training epoch where the lowest validation error occurred (if applicable) and what the lowest validation error was. This addition facilitates error and model monitoring as learning occurs.

Similarly to the existing WEKA MLP, MLP_CS employs an early stopping approach to prevent over-fitting by using a separate validation set of data. As the model begins to over-fit to the training data, the error on the validation set will start to increase. If the validation error continues to rise for a prolonged number of iterations, training is stopped prior to testing the model and the weights of the neural network are returned to the point of lowest validation error.

The modified version the MLP_CS that was used for the studies presented in this paper can be found on the web at ml3.acadiau.ca. While running tests, if the GUI interface is chosen, the resulting interface shows not only the current iteration number and validation set error, but also the lowest validation set error and the iteration at which it occurred.
Table 1. Statistics for the three domains of tasks used in the experiment.

<table>
<thead>
<tr>
<th>Name</th>
<th>Tasks</th>
<th>Primary Task</th>
<th>Secondary Task</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Train</td>
<td>Validation</td>
<td>Test</td>
</tr>
<tr>
<td>Logic</td>
<td>6</td>
<td>13</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>13</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>63</td>
<td>27</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>126</td>
<td>54</td>
<td>20</td>
</tr>
<tr>
<td>Band</td>
<td>7</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>CoverType</td>
<td>6</td>
<td>28</td>
<td>12</td>
</tr>
</tbody>
</table>

The WEKA MLP_CS algorithm has a number of modes of operation that include the creation of a validation set, choice of a specific test set, or alternatively cross-validation using all available training data. Cross-validation was used for all studies presented in this paper and below we explain the changes made for this mode of operation. Additional details can be found at ml3.acadiau.ca.

When using inductive transfer methods, one normally uses an impoverished primary task set so as to demonstrate the value of knowledge transferred from the secondary tasks. Therefore, our testing simulates this condition by proving an impoverished primary task set from which training, validation, and test instances must be drawn. To ensure models are still well developed and generalizable, 10 cross-fold validation is used, as a standard implementation on all WEKA classifiers, and a validation set of 30% is specified. Thus, 10% of the primary set is used for testing, 27% for validation, and 63% for training. Secondary instances, however, are all used strictly for training.

3 Experimentation

The following compares the predictive accuracy of learned models developed under STL using only primary task and csMTL with inductive transfer from secondary tasks. All models are developed with our modified WEKA MLP_CS. The objective is to develop models for the primary task from a small set of training examples so as to observe the effect of inductive transfer from secondary tasks.

Two of the domains, the Band domain and the Logic domain, are synthetic data sets [4]. The third domain, the Covertype set of tasks are real-world data from the UCI repository [4]. Table 1 shows the following statistics for each of the domains: the number of tasks in the domain, the number of input attributes, the number of primary task training, validation, and test set examples, and the number of training examples for each secondary task.

STL and csMTL networks were configured using four-layers of nodes. The input layer receives the primary and context input values for each
task. The first hidden layer contains 20 nodes with sigmoid transfer functions for the Logic domain and 30 nodes for both the Band and the CoverType domains. The second hidden layer contains only one sigmoid node. This second hidden layer is necessary because it provides the shared internal representation between the first hidden layer and output layer. Also, as the networks were configured to generate nominal output, which has two possible classes. Therefore, the fourth layer, which is the output layer, contains two nodes, each representing a separate class. For instance, for the Logic domain we use a 16-20-1-2 network. For each network, the learning rate is 0.001 and the momentum term is 0.9. All tests are run for 50,000 iterations. The study compares the predictive accuracy over all examples using 10-fold cross validation approach.

It is important to note that, under csMTL, the number of training examples for all tasks must be the same so as to ensure that each task has equal opportunity to update the network weights. This requires that the primary task training examples to be duplicated until their number equals that of each secondary task.

Figure 1 shows the comparative results for the three domains from the cross-validation runs. The respective $p$-values from a difference of means $t$-Test are: Band domain, $p=0.716$; Logic domain, $p=0.171$; CoverType, $p=0.036$. These results are not as supportive of the csMTL method as compared to previous results using alternative MLP neural network implementations.

4 Conclusion and Future Work

This paper has described modifications to the WEKA machine learning suite [7] that allows it’s MLP algorithm to accept csMTL encoded data. This modification allows WEKA users to experiment with inductive transfer from related secondary tasks when faced with few training examples for a primary task. Our intention is to increase the availability of inductive transfer machine learning systems to students, researchers
and practitioners. Currently, the work is in progress. The results of these initial experiments show that the WEKA csMTL MLP algorithm is not transferring knowledge from secondary tasks to the primary task at the same level of performance as seen in prior studies using alternate neural network systems. Our next step is to investigate the reason why the current implementation is not working as well as anticipated. There may be a problem with the base MLP code, or perhaps our new approach to cross-validation is awed. In the future our intention is to use the WEKA csMTL MLP as the basis for exploring new theory in inductive transfer and applying the approach to a variety of concept and real-valued tasks.

References