State Interpretation for Social Hierarchical Learning

George Wildridge, Olivier Mangin, Alessandro Roncone, Brian Scassellati
Social Robotics Lab, Department of Computer Science
Yale, New Haven, CT

Science Research Program

2016

Abstract

As robots begin to transition into our everyday lives, there is great potential for robots to collaboratively aid humans in the completion of everyday tasks. Social Hierarchical Learning is an area of active research within machine learning that aims to enable effective human-robot collaboration. This is accomplished through extending the capabilities of hierarchical learning. The long term goal of this research is to enable a robot to transition from student to peer to teacher. In other words, from having no knowledge about a task, to understanding a task well enough to function effectively as a peer, and finally for a robot to be able to teach a task to other humans and robots. Building towards this goal, first and foremost, requires the robot to be able to understand and interpret the environment it is placed in so as to help guide its own actions and to gather information about the task. Utilizing machine learning, both the task and the environment can be visually understood by the robot. Results suggest that utilizing a support vector machine on top of image processing techniques will yield the robot the greatest understanding of the environment.

Task Structure

The chosen task for measuring a robot's transition between learner, peer and teacher is a play set called snap-circuits. Snap-circuits allow for varying levels of task complexity and abstraction. Further, the multiplicity of colors and the labels associated with each piece provides a strong framework for visual recognition. In this study, goals, like make the light bulb light up or make the fan spin, can be set that a human and robot would be asked to work towards together.

A robot's understanding of the task environment can be defined as understanding what pieces are on the board, where they are located on the board, and each piece's respective orientation. Obtaining this information from a picture is challenging as it requires a robot to learn to differentiate pieces based on their shape and labels; a feat generally accomplished through extended training on tens of thousands of images. However, within this study, no previous dataset existed, forcing the creation of dataset of a few thousand images.

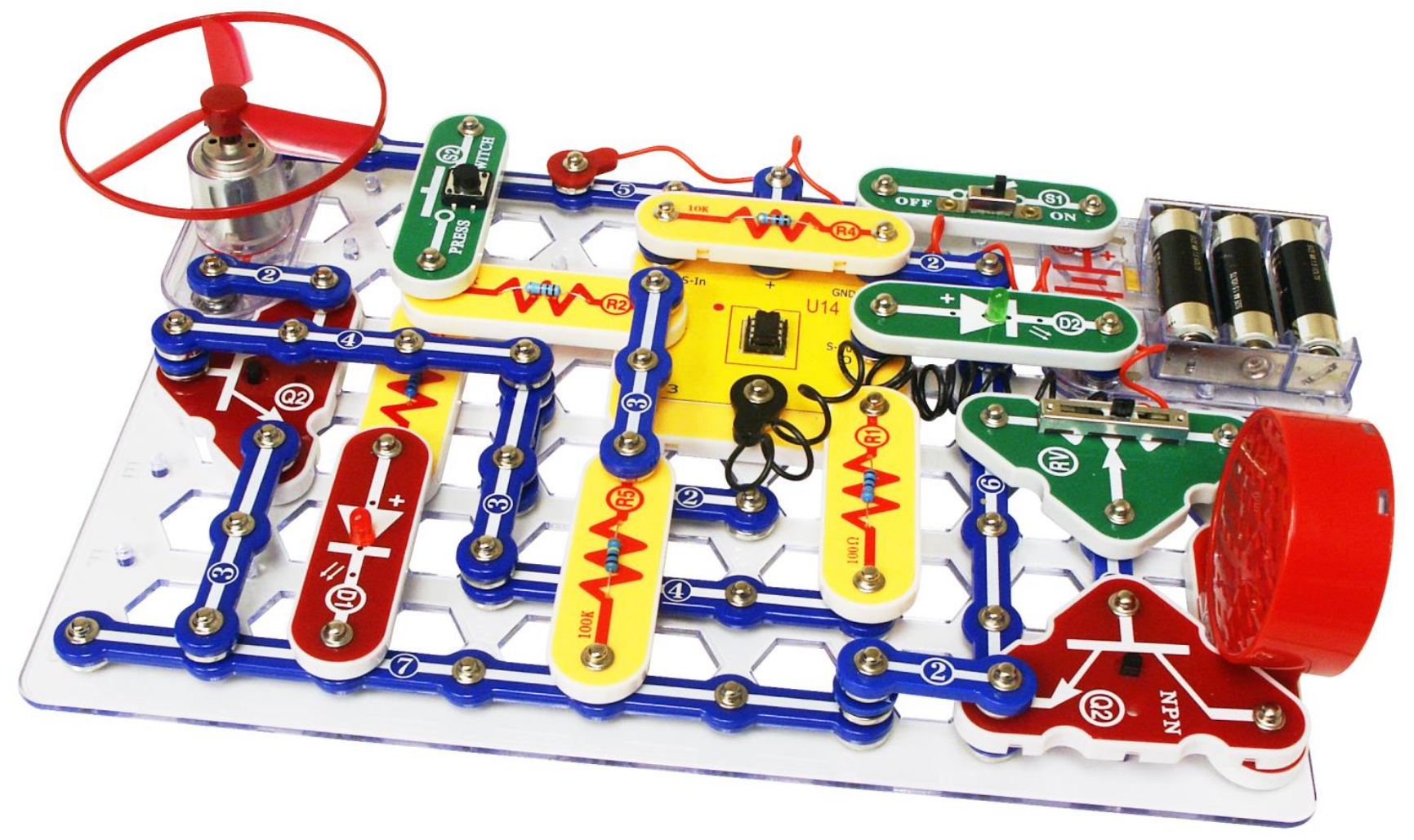

Figure 1. An image of the snap circuits board. This game allows for the complexity and abstraction of a given task to be modulated.

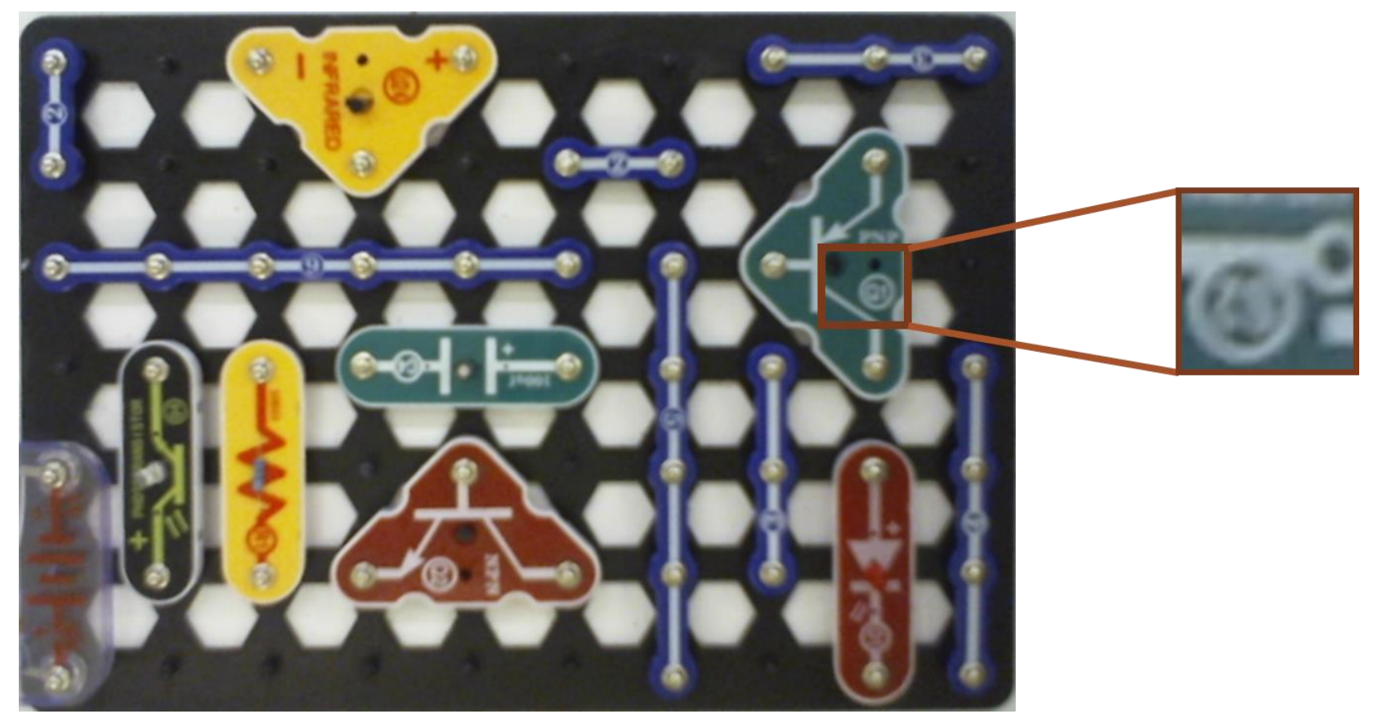

Figure 2. The image of the snap circuit board would be broken up into smaller images as shown above before being fed to the classifiers.

Image Classification

Image classification uses machine learning to separate images into different categories, or classes. In this study there were 32 distinct classes (one for each part). Additionally, machine learning is also used to label part orientation, so each part was also labeled with one of four more classes (north, south, east, west). Three different machine learning techniques were checked against the dataset: a support vector machine, a two layer neural network with a softmax classifier and a convolutional neural network. The basis for all three is a linear classifier. A linear classifier involves an input (an image in this instance) and a set of initially arbitrary parameters known as weights:

$$f = Wx$$

Where x is the image's pixels (stretched into rows) and W is a matrix with as many rows as there are dimensions in the images and as many columns as there are classes. The result of the two being multiplied together is a score for each class. Basically a weighted sum of all the pixel values for each score is being computed. Ideally when an image is passed through, the correct class will have the highest score; however, this is almost never the case. To solve this problem, a classifier, such as a support vector machine or a softmax classifier, are used to interpret the scores and quantify how poorly the linear classifier is working.

Support Vector Machine

Using the output of a linear classifier given random weights, the intent of the SVM is to gauge the degree of how wrong the classification is. It does this by repeatedly finding the difference between the lowest score and the correct score for each image. With the output we can identify how to change the linear classifiers weights in order to improve classification on its next pass. This can be represented by this equation:

$$L_i = \sum_{j \neq y_i} [\max(0, w_j^T x_i - w_{y_i}^T x_i + \Delta)]$$

Where Li is the loss for a single training example, x is a vector of images, w is a vector of weights, y is a vector of labels and delta represents the safety margin. The max is taken in order to clamp possible loss at zero so negative numbers will not have a bearing on the overall loss of the function. If negative loss was allowed to pass through, it would decrease the overall loss which can distort the overall loss and make it seem like we are closer to identifying the correct scores even if we aren't. For the overall loss we must average the loss over all the images:

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i$$

Where N is the number of training examples.

Softmax Classifier

Again using the scores outputted from the linear classifier given random weights, the softmax classifier interprets the scores as the unnormalized log probabilities of the classes. So scores are exponentiated, and normalized before the –log is taken to retrieve the probabilities of the classes. We then attempt to maximize the log likelihood and minimize the -log likelihood of the true class. This can be represented by the following equation where L_i is the loss for a single example, f is the array of class scores for a single example:

$$L_i = -\log\left(\frac{e^{f_n}}{\sum_j e^{f_j}}\right)$$

Neural Networks

Looking at the equation of a neural network is probably the best way to understand what exactly it is. Here's an equation for a 3 layer feed forward neural network:

$$f = W_3 \max(0, W_2 \max(0, W_1\, x))$$

Now a two layer:

$$f = W_2 \max(0, W_1\, x)$$

Finally a linear classifier:

$$f = \mathrm{W}\, x$$

It is a matter of embedding linear classifiers within each other, allowing for multiple sets of weights which in turn means more detail stored about each image. A convolutional neural network separates itself from a feed forward neural network in that the images are initially subsampled before being passed through the neural network. This process allows for more data to be obtained from each image.

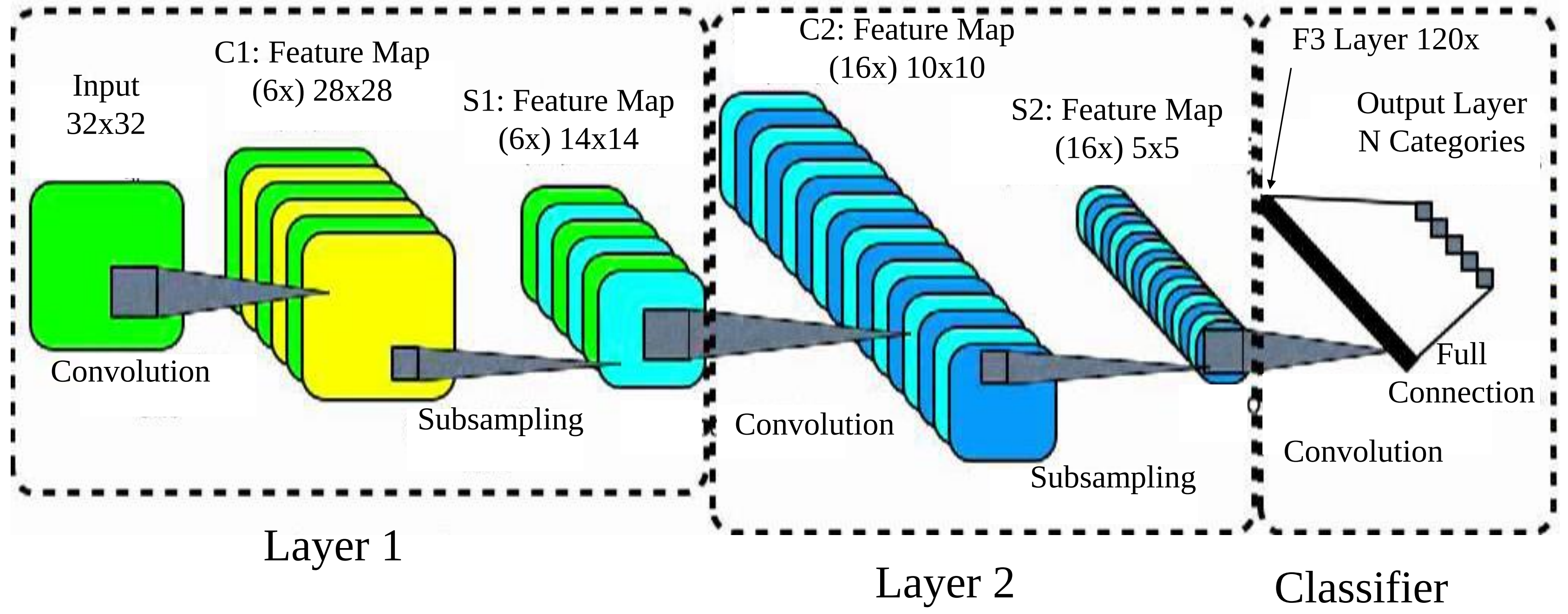

Figure 3. A graphical model of a convolutional neural network with a fully connected classifier. Full connection represents a feed forward neural network. In this study, the classifier makes reference to the aforementioned softmax classifier. The model gives an example of a two layer convolutional neural network; however, depending on the problem, more or less layers can be used.

Results

In testing, separate classifiers were run on both the vertical and the horizontal pieces. In the first phase of testing, an SVM, a two layer neural network with a softmax classifier, and convolutional neural network were run directly on the images as seen on the chart below. Notably, the data shows that the classifiers performed around 90% accuracy on all the data. However, to understand what this means, the difference must be drawn between labeled and unlabeled data. Labeled data contains an ID tag, as shown in figure 2, while unlabeled data does not. The goal of the classifiers are to give the labeled data the correct ID. To gauge how well the classifiers did at this task, the third and fourth columns of the charts must be looked at. Disappointingly, most of the classifiers performed around 0% in classifying the labeled data. Further most of the classifiers predicted that all of the data. Further, most of the classifiers predicted that all of the data was unlabeled. Although this is certainly not the ideal result, the data shows that the horizontal SVM made significantly more progress in classifying the images then either the softmax classifier or the convolutional neural network. During phase two of testing, an effort was made to further increase accuracy by creating a classifier whose sole purpose is to separate a labeled image from an unlabeled image (shown as the label classifier). This then allowed a separate classifier to only distinguish between labeled images(shown as identity classifier). Mixed results were again received. In looking at how well the classifiers performed across the SVM, softmax and convolutional neural network, the classifiers routinely labeled all the data one class as shown by most of the classifiers never straying from the average. The results of phase two again point to the SVM as being the most promising classifier to solve this problem. However, the lack of significant improvement points to a necessity to either incorporate many thousands more images into the dataset, or take a different approach to labeling through more fundamental image processing techniques like contour and color detection. The problem boils down to there being too few labeled images, and too many unlabeled images for the classifier to make meaningful strives in accurately classifying the dataset. Initial steps have been taken in incorporating the more fundamental image processing techniques and, along with a support vector machine, accuracy is expected to rise above 80%.

Phase 1	% Correct	% Labeled Correct	% of Labeled Predicted to be None
SVM Vertical	86%	0%	100%
SVM Horizontal	90%	50%	0%
Softmax Vertical	90%	0%	100%
Softmax Horizontal	90%	0%	100%
Conv Net Vertical	90%	0%	100%
Conv Net Horizontal	90%	0%	100%

Phase 2	Classifier	% Correct	% Labeled Correct	% Strayed From Average
SVM Vertical	Label	5%	100%	0%
	Identity	0%	0%	100%
SVM Horizontal	Label	3%	100%	0%
	Identity	50%	50%	100%
Softmax Vertical	Label	90%	0%	0%
	Identity	0%	0%	0%
Softmax Horizontal	Label	90%	0%	0%
	Identity	17%	0%	0%
Conv Net Vertical	Label	90%	0%	0%
	Identity	0%	0%	0%
Conv Net Horizontal	Label	90%	0%	0%
	Identity	0%	0%	0%

Future Work

Future work will focus on further improving the robot's visual understanding of the environment using a support vector machine on top of image processing techniques such as contour and color detection. Focus will then shift to applying artificial intelligence techniques like learning from demonstration and hierarchical learning to enable a robot to shift from learner to peer to teacher.

Acknowledgements

I would like to thank the Social Robotics Lab at Yale University, Dr. Alessandro Roncone, Dr. Olivier Manguin, Dr. Bradley Hayes and Prof. Brian Scassellati for both the tremendous opportunity and their assistance in understanding the material. I would also like to acknowledge Dr. Christopher Hogue and the Science Research Program for their support.