Using Artificial Neural Networks to Model Siegler's Balancing Task

Tyler W. Zon and Benjamin F. Xie

2014

Abstract

Understanding balance is a necessary part of a child's learning experience. Siegler found that children often go through four stages in understanding balance and torque. Computer simulated models of Siegler's experiment have been created using Artificial Neural Networks. Shultz used a neural network to model Siegler's balance task and found that the network moved through the four stages argued by Siegler. Whereas Shultz had a very formal definition of each stage, we believed that these definitions were arbitrary; instead, we focused on observing the neural network's incremental learning by avoiding the explicit definitions. Five experiments were performed with three categories of problem type (easy, medium, and hard). Each of these problems represent a subset of the balance scale problems used in Siegler's experiment. We found we were able to replicate the stage-like development and model the whole developmental process of learning the balance task.

1 Introduction

Balancing and being able to understand balance are fundamental abilities that people need in everyday life. Children are not born knowing how to balance; cognitive psychologists have studied how humans learn to understand this concept [12, 13]. One important component in a comprehensive understanding of balance involves applying the concept of torque. In scientific terms, torque describes the cross product of displacement and force which normally produces a rotation. It describes how distance from a pivot point or fulcrum impacts the rotation of an apparatus. Children must eventually learn that weights put further away from the pivot point have more torque than weights that are closer to the fulcrum.

Psychologists have employed many different strategies for studying the process of learning balancing in children [4]. One of the more notable examples, Siegler's discretized balance experiment, (Figure–1) has been frequently employed by researchers in their experiments. The discretized balance experiment puts four weight slots on each side of a fulcrum on a scale. Siegler created six different types of problems to test children's balancing abilities. In the experiment, children were presented with the balance scale (Figure–1) and were asked which side the scale would lean if there were no supporting mechanisms.

Siegler interviewed many kids between the ages of 5 and 17 and noticed that there were 4 stages of development that emerged. In the first stage, children are able to predict outcomes of the balance experiment solely based on weight information. In stage two, children use weight information and begin to use distance information, forming a basic idea of torque. In stage three, children use both distance and weight information independently but get confused when they conflict. In stage

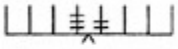
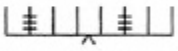
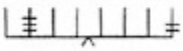
PROBLEM TYPE	RULE I	II	III	IV
Balance 	100	100	100	100
Weight 	100	100	100	100
Distance 	0 (Should say "Balance")	100	100	100
Conflict-Weight 	100	100	33 (Chance Responding)	100
Conflict-Distance 	0 (Should say "Right Down")	0 (Should say "Right Down")	33 (Chance Responding)	100
Conflict-Balance 	0 (Should say "Right Down")	0 (Should say "Right Down")	33 (Chance Responding)	100

Figure 1: Siegler diagnosed 4 main stages in human development when it comes to learning to solve balance problems. He tested children on six types of problems. Balance problems are completely balanced in terms of position and weight. Weight problems have differing numbers of weight in the same relative position on each side of the fulcrum. Distance problems place weights in different relative locations on either side of the fulcrum. The conflict types of problems are where balance, weight, and distance problems might predict the scale to lean to one side but because of torque the balance scale performs differently [12, 13].

four, children have a full grasp of torque and are able to perform correctly on a number of balance problems.

In addition to the developmental stages suggested by Siegler, Ferretti and Butterfield have been able to show that children are able to perform well on the balance task if there are large differences in torque on either side of the scale [3]. This was coined the torque-balance effect. This suggests that the more obvious the difference in torque on either side of the fulcrum the easier the problem becomes to the child. Researchers have postulated that distinguishing this critical information allows the child to perform better on balancing problems. It has also been noted that the balance task can be grouped into a set of problems that require multiple dimensions of understanding. Children often struggle with these problems until they are able to understand how the dimensions work together. Some of these problems include the ideas of fullness, inclined planes, projection of shadows, and conservation.

Machine learning techniques and connectionist models have been implemented to learn Siegler's balance problem [8]. Newell and Langley used balancing rules based on a look-ahead search mechanism and discrimination learning, respectively [7, 5]. Both researchers were unable to achieve stage four of Siegler's stages due the inability of the rules to represent torque in an effective manner.

Artificial Neural Networks (ANN's) have been trained to correctly assess some balance problem. ANNs are used in supervised learning and involve training a network of nodes and weights to produce correct output for given input; error is calculated at the end of each input and is used to adjust the weights via the "back-propagation" algorithm. McClelland was able to achieve the first three stages of Siegler's experiment using such a back-propagation network, but could only do so by creating a training set that is loaded with weight and balance problems [6]. McClelland also had to alter the structure of the neural network to separate both distance and weight information, to allow it to reach stage three of Siegler's stages. McClelland's claim was that children spend a lot of time in these stages experimenting with both weight and distance information so the bias seemed necessary.

Schmidt and Shultz used McClelland's network to show that it is possible to achieve stage four using his model; a downside, however, was that the ANN would be unable to produce correct answers for stages 1 and 2 [10]. Schmidt and Shultz progressed to a cascade correlation model that would allow a back-propagation network to recruit hidden units systematically and learn in a style called batch learning [11]. Batch learning is when the training set is divided and learned in parallel and was a necessary condition for the implementation of the cascade correlation algorithm used by Schmidt and Shultz [9]. Their implementation was able to proceed through all four of Siegler's stages. This was shown by a similar level of error to the children sampled by Siegler in the ANN used by Schmidt and Shultz.

Schmidt and Shultz diagnosed stages based on the combination of correct answers by a neural network in a series of 24 questions. A diagnosis of stage 4 meant 20 of the 24 problems were answered correctly. Stage 3 was determined if the neural network responded correctly on 10 out of 12 balance, weight, and distance problems, and responded correctly in less than 10 of the 12 conflict type problems. Stage 2 meant the neural network answered at least 13 out of 16 correct on the balance, weight, distance, and conflict-weight problem but less than 3 out of 8 on the conflict-balance and conflict-distance problems. Stage 1 was determined if the network could correctly answer 10 out of 12 weight, balance, and conflict-weight problems but could only answer correctly 3 out of 12 of the distance, conflict-balance, and conflict-distance problems [2, 1].

These rigorous definitions of stages do not seem to mirror the experimental process most children go through to completely understand balance. Schmidt and Shultz focus more on finding a definitive stage progression, despite noting that most children move through U-shaped development or alternating between stages to finally reach the next stage. In our experiment we wanted to address McClelland's unsuccessful attempt to model the balance task with a backpropagation neural network. We also wanted to exhibit a more fluid version of stage like growth than Schmidt and Shultz' rigorous rules. In our experiment, we use Siegler's balance scale to show that not only can progressive stage-like development be seen in all problems, but also a neural network is able to learn the balance scale task relatively quickly. We wanted to see how a neural network would perform on each of the problem types and how the research in human developmental psychology would compare with the development of the ANN. Our main hypothesis would be that the training would appear to go through stages corresponding to human development. The problem types that humans learn first would also be what the network would learn first.

2 Experiments

In order to test the hypothesis that ANNs would go through similar stage-like development as humans, a neural network was configured. A basic neural network with a fixed topology like the one McClellan used in his experiment was chosen. The network took 8 inputs that corresponded to weights on each peg of a balance scale. "1 0 0 0 0 0 0 1", for example, represents a balanced scale with 1 weight on each end of the scale. "2 0 0 0 1 0 0 0" corresponds to a 2 weights on the leftmost position on the scale and 1 weight on the right-side peg closest to the center. This configuration would be left leaning. The outputs of the ANN would be either left-leaning, balanced, or right-leaning. This was represented as three outputs: "1 0 0", "0 1 0", and "0 0 1", respectively. This allowed us to more effectively use the sigmoid activation function, as weights in the network normally tend toward either 0 or 1. A hidden layer of 5 nodes was used to allow more complex behavior to evolve. Similar to Schmidt and Shultz a learning rate of 0.25 was used because there were balanced configurations in our training set.

In order to track stages, we divided the six different problem types into three problem categories: easy, medium, and hard. The easy category included balance and weight problems, the medium category included distance and conflict weight problems and the hard category contained randomly assigned weights. It was postulated that these categories would allow us to see stage-like development as well as the transitions between stages unlike Schmidt and Shultz. We also took issue with the diagnosis of stage 4 in Shultz experiment. A person who was in stage 4 should be able to understand a more complicated configuration of haphazardly placed weights placed on the balance scale. For this reason, both the conflict-balance, and conflict-distance were removed as problem types. We tracked two statistics as the network underwent training. First, we kept track of the Total Sum Squared(TSS) error throughout each epoch. Second, we tracked the percentage of problems the network was getting correct for each problem type.

In our setup program, we generate a set of 1000 input strings representing each scale balance problem. We also generate a corresponding list of 1000 targets and problem difficulty markers. The problem difficulty file allows us to keep track of what problem types the neural network is learning. We can thus plot out this data to see possible stages of learning. We originally split the input file to have an even number of easy, medium, and difficult problems. Within each category, we have an equal chance to generate any problem types that fall into that difficulty category. For example, the easy category can generate roughly equal numbers of balance and weight problems.

In our first experiment, we ran the training on the data as given. In the second experiment, we remove the balance problem in order to see how this might affect the data. In our third experiment, we split the easy problem into the constituent weight and balance problem; we further changed the inputs to the network from between 0 to 3 to between 0 and 1.

The final experiment we performed involved changing the input setups once more—in addition to having roughly equal numbers of easy, medium, and hard problems, we also changed the setup to have roughly even numbers of right-leaning, left-leaning, and balanced outputs (Table 1).

3 Results

In the first experiment, we found that the easy problems seemed to be actually more difficult than the others. As can be seen in Figure–2, the easiest problem type ended up being harder to learn.

Experiment	Treatment	Input range
1	All problem types included	0,3
2	Balance problem removed	0,3
3	Scaled inputs down and separated Easy group into balance and weight problems	0,1
4	Ensured a roughly even number of balanced, left-leaning, and right-leaning problems	0,3

Table 1: This table briefly summarizes each experiment in terms of the treatment applied and the input range of the values used.

In the zoomed in graph of the first five generations, all problem types are learned at similar rates until roughly 1000 steps in. At this point, there is a clear difference between the easy problem types with the other difficulties. By the end of the 80 epochs, however, the network manages to learn all the different problem types just as people do.

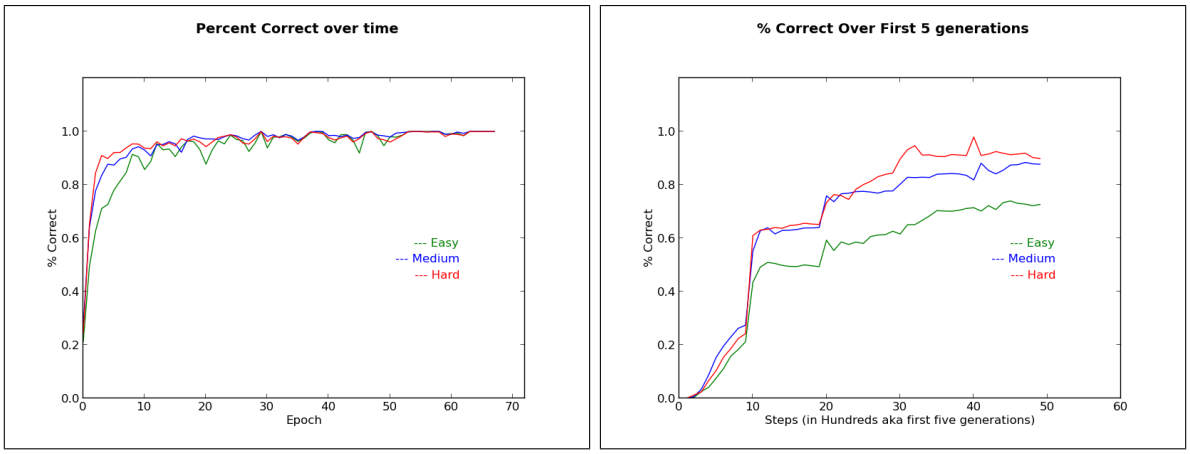


Figure 2: The first experiment which completed roughly on the 70th epoch. On the right is a closer look at the left graph. We zoom in on the first five generations as most of the learning seems to be happening here.

After examining the results more closely, we came up with the hypothesis that the balance problem was causing the easy experiments to be really hard. In experiment 2 seen in Figure–3, the balance problem was removed and the training was ran again. As expected, the easy problem became the first problem type to be solved by the network. It was not anticipated, however, how strong of an effect the balance problem had.

In the last two experiments, we brought the balance problem back as we felt it to be an essential part of learning the concept of balancing. In the third experiment, in Figure–4, we split the easy problem into its constituent problems so we could see the results for each type of easy problem. In this experiment we scaled the input down to be between 0 and 1 instead of 0 to 3 for each peg. This minor change was to see whether the sigmoidal activation function's tendency toward 0 or 1 would

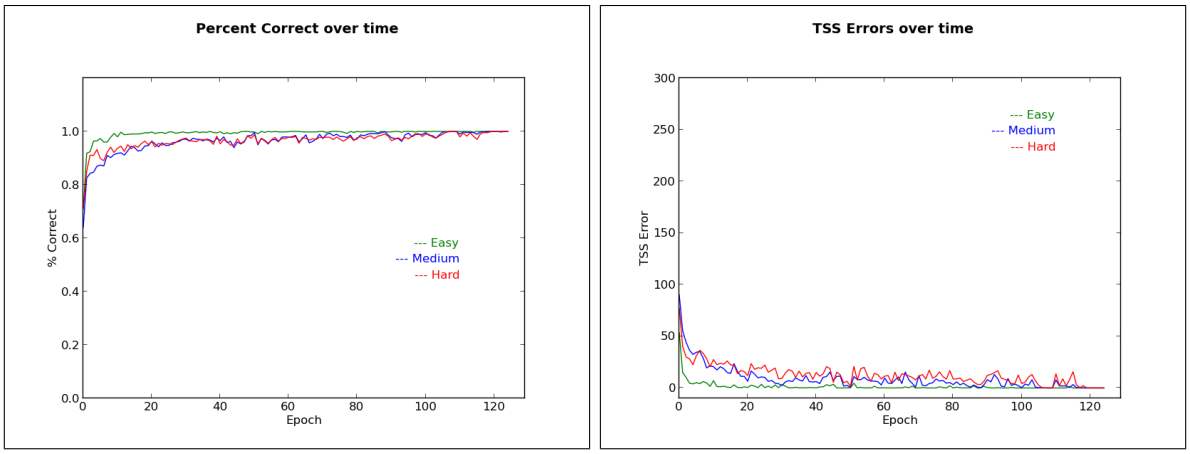


Figure 3: In experiment 2, we remove the balance problem so easy problems consisted solely of weight problems. On the left is the percent correct over time for the different problem types. The right shows the error over time.

be hurt by our inclusion of inputs greater than 1. We thought that perhaps this might partially be responsible for the difficulty in solving the balancing problem. Oddly enough, the training as a whole took much longer. We cut it off at epoch 100 as the data stops being interesting fairly early on. The data never perfectly converges as it did before, instead fluctuating near 100% correctness. As the difficulty of balance problem was mostly unaffected, we decided to change the inputs back to being between 0 and 3 for the final experiment.

For the final experiment, shown in Figure–5, we tried to address the possible issue that there was an uneven number of left-leaning, right-leaning, and balanced examples. Typically, the balanced targets would be much fewer in number than either of the left or right-leaning outputs. We increased the balanced problems by making hard problems have an even split between balanced and unbalanced hard configurations. The hard problems ended up being much harder, and easy problems appeared as expected to be easier.

4 Discussion

Through our experiments, we found that basic back-propagation neural networks were able to learn Siegler's balance task with relative ease. Our results showed some implicit stage-like growth, representing the way children develop in learning the balance problem. Although the stages did not correspond directly with what humans go through (hard problems were not always hardest to solve), we still observed that some problems were clearly harder to learn than others. Each of our experiments helped us better understand the roles of connectionist models in learning the balancing problem.

In our experiments, the balance problem was always the hardest problem type. Despite being classified as one of the easiest rules to learn in Siegler's research, both the percent correct and TSS error showed that the balance problem was the toughest. When we removed the balance problem from the easy group, our results showed more stage-like growth. We postulated that our experimental setup was favoring configurations that were unbalanced but even after modifying our

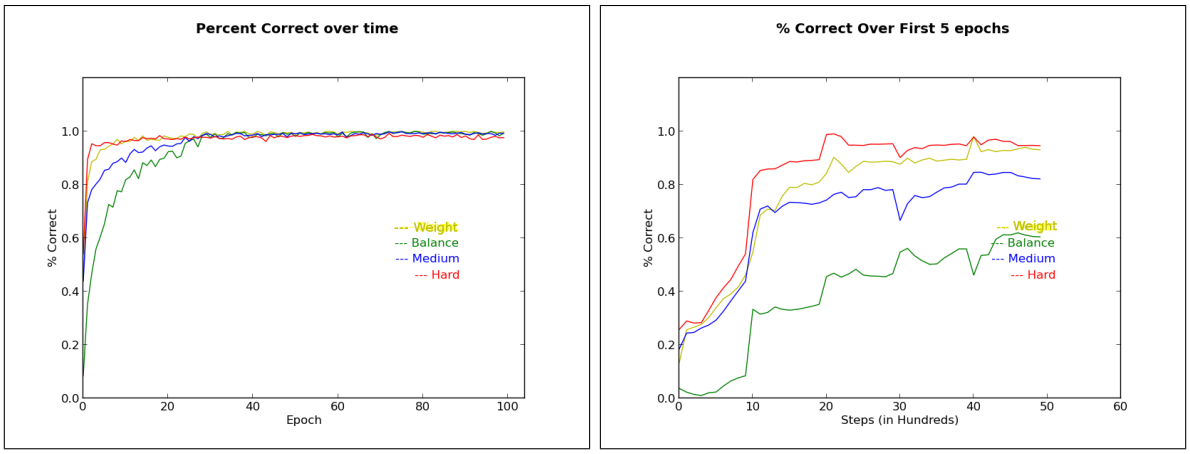


Figure 4: This experiment involved scaling the inputs down to between 0, 1. The results were very similar to the original experiment; the only difference was that the whole problem was never fully learned within 500 epochs. We reduced the number of epochs to only 100 as the results don't change much after this point.

setup the balance problem still remained the most difficult. One potential reason for this result is that perfect balance and symmetry within inputs are intrinsically more difficult to learn for neural networks. Often networks have trouble distinguishing completely symmetrical input because fully connected networks are somewhat unstable—each weight has to capture so many different elements of the problem. This means a slight change in input type would cause a large difference in the network [1]. Because both simple and complicated balance problems exist, the network might be getting confused. Another possible reason for the difficulty of the balance problem is the torque difference effect. Because there is no difference in torque the network could have a harder time learning the problem. Nevertheless, after a while, the full balance problem is eventually learned and can be fully verified on a test data set of totally new values.

Furthermore, we noticed a small difference between medium and hard problems. Because hard problems were randomly configured, and weights could be placed in multiple locations only in these types of problems, we believe the neural network was able to quickly classify this group of problems. This allowed the neural network to more quickly learn these types of problem.

Though we intended to show a more gradual development with both stages and transitions, it was difficult to fully diagnose stages based on our classification. While error decreased in all of our problems (easy, medium, and hard), the error was decreasing in all types of problems at the same time. Rather than seeing the clear cut four stages of development, the growth we saw was more incremental as we intended. Shultz and Siegler diagnosed stages based on correct answers but did not look at the incremental learning that takes place on all problems; the reason they saw well-defined stages is due to how they defined their stages.

Numerous experiments could be performed in the future to better understand and represent Siegler's balance experiment. Schmidt and Shultz' cascade correlation is very similar to NeuroEvolution of Augmenting Topologies (NEAT). NEAT is a genetic algorithm that complexifies a neural network as it moves through generations. NEAT also allows for speciation and crossover of the most successful versions of the neural network. NEAT might be worth using as the complexification

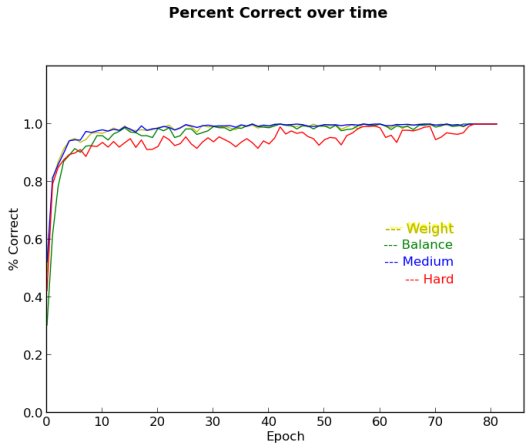


Figure 5: In our final experiment, the hard problems ended up being harder to learn. This is likely due to balanced problems being intrinsically more difficult to solve.

may create a better network topology than the fixed topology used in this experiment. Another experiment that could be performed is to mirror the exact types of problems in Siegler's experiment but maintain our version of categories; this would mean implementing ways to generate the various conflict problem types. We believe this might show even more of an incremental learning strategy than the discrete stages suggested by Siegler. Furthermore, it would be interesting to move away from Siegler's discretized experiment to a continuous balance scale to see if the torque difference effect played an even larger role.

References

[1] P Cunningham. Stability Problems with Artificial Neural Networks and the Ensemble Solution. *Trinity College*, 2000.

[2] Fredric Dandurand and Thomas R. Shultz. Modeling Acquisition of a Torque Rule on the Balance-scale Task. *published online*, 2009.

[3] R.P. Ferretti and E.C. Butterfield. Are Children's Rule Assesment Classifications Invariant Across Instances of Problem Types? *Child Development*, 1986.

[4] J. Boom S. Kunnen. Rules in the Balance: Classes, Strategies, or Rules for the Balance Scale Task. *Cognitive Development*, 16, 2001.

[5] E. Langley. A General Theory of Discrimination Learning. *MIT Press*, 1987.

[6] J.L. McClelland. Parallel Distributed Processing: Implications for Cognition and Development. *Oxford University Press*, 1989.

[7] A. Newell. Unified Theories of Cognition. *Harvard University Press*, 1990.

[8] P.T. Quinlan and H. van der Maas. Re-thinking Stages of Cognitive Development: An Appraisal of Connectionist Models of the Balance Scale Task. *Cognition*, 2007.

[9] W.C. Schmidt and C.X. Ling. A Decision-tree Model of Balance Scale Development. *Machine Learning*, 1996.

[10] Thomas Shultz and William Schmidt. Modeling Cognitive Development With a Generative Connectionist Algorithm. *McGill University*, 1991.

[11] Thomas Shultz and William Schmidt. Modeling Cognitive Development on Balance Scale Phenomena. *Machine Learning*, 1994.

[12] Robert Siegler. Three Aspects of Cognitive Development. *Cognitive Psychology*, 1976.

[13] Robert Siegler. Monographs of the Society for Research in Child Development. *Cognitive Psychology*, 1981.