Modeling Number Sense Acquisition in A Number Board Game by Coordinating Verbal, Visual, and Grounded Action Components

Arianna Yuan (xfyuan@stanford.edu) **James L. McClelland (jlmcc@stanford.edu)**
Department of Psychology, Center for Mind, Brain, Computation and Technology, Stanford University, Stanford, CA

Abstract

Previous studies including Ramani and Siegler (2008) have shown that playing a number board game improved students performance on several numerical tasks, including numeral identification, magnitude comparison, counting and number line estimation. However, the computational mechanism underlying such number sense acquisition remains unclear. Here, we aim to fill this gap by building a model that simulates play of the game as well as the basic numerical tasks. We hypothesize that cognitive components that are used in the basic tasks are recruited to work together when children play the game, so that the learning induced by playing the game also manifests itself in those tasks. We reproduced the empirical findings with a neural network model implementing our hypothesis. This computational approach demonstrates how a single model that coordinates components of number processing in different modalities (visual, language and spatially-guided action) can explain the number sense acquisition in number board game playing.

Keywords: Numerical Cognition; Mathematical Education; Neural Networks; Board Games

Introduction

Mathematical concepts are notoriously hard to learn, perhaps because they often involve a range of distinct properties. Even the seemingly simple mathematical concepts, such as the natural numbers, may have diverse properties and multiple representations. For instance, the concept of "five" can be grounded as the *cardinality of a set*; as a *position on a line*; as a *distance in space*; or as a *number of events in a temporal sequence*. In fact, researchers have summarized these observations and proposed that humans use several grounding metaphors to understand numbers (Lakoff & Núñez, 2000), including *arithmetic as object collection*, *arithmetic as the use of a measuring stick* and *arithmetic as motion along a path*.

Given these diverse groundings, the perceptual variance of natural numbers may be much larger than that of many ordinary concepts. For example, "five dogs", "five houses", and the position in a row between 4 and 6 are perceptually very different, yet they can all instantiate the number "five". How do children learn to link different representations of numbers, particularly non-symbolic ones, such as cardinality and distance in space, to symbols such as verbal number words and written Arabic numerals? This problem is called the symbol grounding problem (Harnad, 1990), thought to be equivalent to the problem of determining how we assign a meaning to a symbol (in our case, a number word).

Many researchers have attempted to provide an answer to this problem. One popular account, the *approximate number system (ANS) mapping account*, assumes that a symbol acquires its numerical meaning by being mapped on a non-verbal and innate ANS. Evidence supporting this hypothesis includes longitudinal studies in developmental psychology. For instance, there is a large body of literature showing a correlation between non-symbolic number processing and symbolic math (Halberda, Mazzocco, & Feigenson, 2008; Libertus, Feigenson, & Halberda, 2011) and arguing for a causal link between the two (Park & Brannon, 2014).

Whatever the origin of non-symbolic number may be, the question remains, what is the process whereby the many diverse aspects of non-symbolic number and symbolic numbers are acquired, to support skills such as number-space mapping and numerical magnitude comparison? In the current paper, we begin to address this question. Particularly, we will focus on a number board game that has been used in several studies to provide a rich learning environment that grounds various aspects of numerical processing and links them to the printed and spoken symbols for numbers. In the seminal paper by Ramani and Siegler (2008), the authors showed that playing this number board game for roughly an hour spaced over several sessions increased low-income preschoolers' proficiency on four diverse numerical tasks: numerical magnitude comparison, number line estimation, counting (defined as reciting the count list from 1 to 10) and numeral identification. Below we describe the details of their intervention study.

In the board game, the board includes 10 horizontally arranged squares of the same size, with the word "Start" at the left end and "End" at the right end. There are two conditions in the study. In the experimental condition, the board the squares are numbered consecutively from 1 to 10 in order from left to right. In the control condition, the squares only have alternating colors. In addition, in the experimental condition, the game has an associated spinner with a "1" half and a "2" half, whereas in the color board version it has a spinner with colors that correspond to the colors of the squares on the board. Before playing the game, the participants were tested on 4 numerical tasks: numeral identification, magnitude comparison between two numbers, counting and number line estimation.

In the game, children chose an animal token and used it to mark their progress on the board. Children were instructed to take turns spinning the spinner and were told that the one who reaches the end first wins the game. Children in the experimental condition were told that on each turn, they would move their token the number of spaces indicated on the spinner. Also, they were required to say the number that they spun and the numbers on each of the squares they reached as they moved. For instance, if they were on 5 and they spun a "2", they would first say "two" then say "six", "seven" as they moved. In the control condition, children were told that

they would move their token to the next square with the color that matched the one they spun. Similar to the experimental group, they need to say the color they spun and the colors on the squares they reached as they moved.

After the participants played the game several times in each of four short sessions over the course of several weeks, they were tested a second time on the 4 numerical tasks mentioned previously. The experimental group demonstrated significant improvement, whereas the control group did not. The gains remained 9 weeks later in a follow-up session, in which the same tasks were tested a third time.

Here we propose a mechanistic account for the enhancement of numerical processing skills that was induced by playing the game. We hypothesize that multiple cognitive components (visual, language and spatially-guided action) are recruited and coordinated in the game environment. Particularly, the process of moving the token incrementally along the number board (motor) is coordinated with saying the next number word through the count list (verbal) as well as naming the printed numeral on each square tile on the board (visual). We suggest that the various components engaged in this process are also engaged in the basic numerical tasks, allowing learning occurring in the game to transfer to the basic tasks.

Another motivation for the current paper is that we want to address one of the common shortcomings of neural network models, which is their lack of flexibility in multi-task learning scenarios. In the current paper, we would like to show that as long as the model is composed of meaningful cognitive components that are also used in other tasks, it will be possible to demonstrate that training on one task could result in improvement on other tasks. The idea of constructing neural networks with multiple components that are responsible for different sub-tasks aligns with some recent advance in machine learning, e.g., neural networks composed of distinct modules have been used to solve language grounding problems (Andreas, Rohrbach, Darrell, & Klein, 2016; Johnson et al., 2017).

Below we first describe the architecture of our model, followed by the experiments we ran to simulate the learning in the number board game and the resulting learning effect. Finally, we present the implications of our results, some limitations of the current work and some future directions.

Model Architecture

There are three neural network modules in our model: the Visual component, the Language component (Figure 1) and the Action component (Figure 2).

Visual Component

The Visual component is composed of a pre-trained neural network named the ResNet (He, Zhang, Ren, & Sun, 2016) and two fully-connected readout layers. The ResNet is a deep neural network trained to recognize objects in ImageNet, a large image database of natural images with hand-annotated labels (Deng et al., 2009). The ResNet consists of stacked

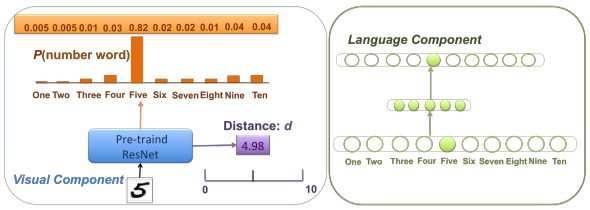


Figure 1: Illustration of the Visual component (left) and the Language component (right).

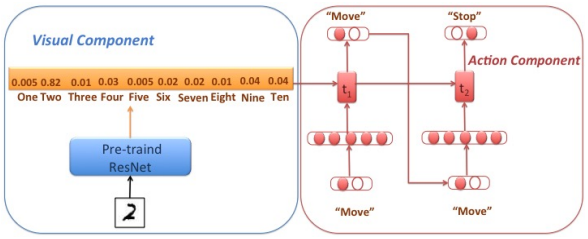


Figure 2: Illustration of the link between the number word output of the Visual component (left) and the Action component (right).

convolutional layers with non-linear activation functions and identity shortcut connections that skip one or more layers (see He et al. for details). We use the ResNet to process each image in our dataset and use the hidden activation of the last hidden layer as the feature vector of the image, i.e., as the image embedding. We then apply two fully-connected readout layers to the image embeddings, one *number word* readout layer and one *magnitude* readout layer. All of the images in our dataset are images of Arabic numerals ranging from 1 to 10 (see the Experiment Section for details). The *number word* readout layer is used to decode the numbers in the images. It outputs a probability distribution over the ten possible number words (one to ten). The *magnitude* layer is used to decode the magnitude of the number represented as a scalar, thought to be provided as a target for learning by the perceived distance of the number's location on the number line from zero. When simulating the number board game, we make the assumption that children attend to the board in two frames of reference, a global one and a local one: in the local frame of reference, they attend to and recognize the digit printed on the current square. In the global frame of reference, they keep track of how far their token has traveled from the "Start" point. These two processes are implemented by the *number word* readout layer and the *magnitude* readout layer, respectively. In the simulation of the number board game playing, these two layers were trained simultaneously. During the training we did not update the weights of the ResNet and only the connection weights of the two readout layers were updated. The equations of the Visual component can be written as follows:

$$
\begin{aligned}
e_i &= \text{ResNet}(I) \\
P(\text{number word}) &= \phi(W_{nw} e_i + b_{nw}) \\
m &= \text{ReLU}(W_m e_i + b_m)
\end{aligned} \tag{1}
$$

where $I \in \mathbb{R}^{28\times28\times3}$ is the raw pixels of the image, $e_i \in \mathbb{R}^{512}$ is the image embedding, $W_{nw} \in \mathbb{R}^{512\times10}$ ($W_m \in \mathbb{R}^{512\times1}$) are the connection weights from the embedding layer to the *number word* (*magnitude*) readout layer, $b_{nw} \in \mathbb{R}^{10}$ ($b_m \in \mathbb{R}^1$) are the biases of the *number word* (*magnitude*) readout layer. The softmax function $\phi(\boldsymbol{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$ is used to normalize the logits of the *number word* readout layer in order to get the probabilities over number classes. The prediction is thus the one with the highest probability. The ReLU activation function is used to restrict the magnitude output to be non-negative, i.e., $\text{ReLU}(x) = \max(0, x)$.

Language Component

This component learns the successor function in counting, i.e., it predicts the next number in a count list given the current number (Figure 1, right). Particularly, during training, we use (current-number, next-number) pairs as data, e.g., ("five", "six"), and the model takes the current-number as input and predict the next-number. During testing, we iterate from zero, and the score is coded as correct up to the point of the first error. The number words are represented as one-hot vectors and they are fed to a fully-connected layer with rectifier activation function (ReLU), which is followed by a fully connected output layer with the softmax activation function. The mathematical formula for this module can be written as:

$$
\begin{aligned}
h_l &= \text{ReLU}(W_l e_w + b_l) \\
P(\text{next word}) &= \phi(W_n h_l + b_n)
\end{aligned} \tag{2}
$$

where $e_w \in \mathbb{R}^{10}$ are the word embeddings for the stimuli (one-hot vector), $W_l \in \mathbb{R}^{10\times10}$ are the connection weights from the input embedding layer to the hidden layer. We use the softmax function $\phi(x)$ to normalize the logits to get the probabilities over the possible number words.

Action Component

The Action component is a recurrent network that learns to output a sequence of "MOVE" actions before it outputs a "STOP" action (Figure 2, right). We use the Visual component described above to read the spinner, which shows either "1" or "2", to generate a probability distribution over all possible number words. This serves as the initial hidden state of the Action recurrent network. The Action network then takes the initial hidden state h_0 and the initial action a_0 to predict the next action, which is either "MOVE" or "STOP"[1], and the new hidden state and the predicted action are used in the next time step t. The mathematical formulae can be written as:

$$
\begin{aligned}
h_t &= h_{t-1} + ReLU(W_a e_a) \\
O_t &= \phi(W_m h_t + b_m)
\end{aligned} \tag{3}
$$

[1]In this board game the first action is always "MOVE", so there's no need to make a prediction for the first action.

Figure 3: Digits stimuli used in the current task. 1 - 9 are randomly selected from the MNIST dataset and the 10s are generated by randomly selecting 1s and 0s from the dataset and combine them (LeCun et al., 2010).

where $e_a \in \mathbb{R}^5$ are the embedding of the action, $W_a \in \mathbb{R}^{5\times10}$ are the connection weights from the embedding layer to the hidden layer. We use the softmax function to normalize the logits of the output layer to get the probabilities over the two possible actions ("MOVE" or "STOP").

Experiments

We use digit images from the MNIST database (LeCun, Cortes, & Burges, 2010), a dataset of handwritten digits with labels. We use a randomly selected subset of the original training data to construct our training dataset, which contains 10,000 samples (Figure 3), and our test dataset contains 10,000 samples that do not overlap with our training data.

Our simulations have two conditions that correspond to the experimental condition and the control condition in Ramani and Siegler's empirical study. In the experimental condition, there are three phases: the pre-test training, the number board game training and the post-test training. For simplicity, and because no numbers were used in Ramani and Siegler's control condition, the simulation control condition included only the pre-test training and the post-test training. In the pre-test training phase, the *number word* readout layer of the Visual component is trained on the numeral identification task for 34 batches, the *magnitude* readout layer is trained on the number line estimation task for 46 batches and the Language component is trained on the counting task for 16 batches. The numbers of batches for each task are chosen to produce the pre-test accuracies that approximate the ones in Ramani and Siegler's original paper.

Each batch contained 100 trials. The numerals were equally distributed in trials of each task. In counting, each trial involved a single transition from a "current-number" to the "next-number". We used backpropogation to train the network. In the board game training phase the models were trained on board game trials corresponding to individual turns in the game for 50 batches. We next describe in detail how all the cognitive components work together in a single trial. Assuming that the agent's token is at square "3" and the spinner yields "2", the modules will perform the following sequence of computations:

1. The Visual component reads the spinner ("2") and feeds the computed P(number word) to the Action component as its initial hidden state.

2a. The agent moves one step forward. The Visual component takes the image of the square where the agent's token is currently located ("4") and recognizes the number

printed on it, i.e., "four".
2b. The *magnitude* readout layer of the Visual component predicts the distance between the token's current location to zero, i.e., 4.0, as the supervision signal is available in the game.
3. The Action component takes the initial hidden state and the embedding of the initial action "MOVE" to predict the next action "MOVE".
4a. The agent moves one step forward. The Visual component takes the image of the square where the agent's token is currently located ("5") and recognizes the number, i.e., "five".
4b. The *magnitude* readout layer of the Visual component predicts the distance between the token's current location and the "Start" point, i.e., 5.0.
4c. The Language module takes the number that the Visual module recognized in the last time step ("four") to predict the current number, i.e., "five".
5. The Action component takes the last hidden state and the last action "MOVE" to output the next action "STOP".

To summarize, in this single trial, the Visual component needs to map the image of "4" to ("four", 4.0) and "5" to ("five", 5.0); the Language component needs to map "four" to "five" and the Action component needs to map the image of "2" (the spinner's output) to the action sequence "MOVE (given), MOVE, STOP". On trials where the spinner's output is 1, then only the steps 1, 2, 5 will be performed.

Finally, in the post-test training phase, in both conditions the neural modules are trained on one batch to simulate the learning experience gained during the 9 weeks between the immediate post-test and the follow-up test. This is termed as the "1-batch post-test training" simulation. This amount of training yields changes in performance that are comparable to the change from the post-test to the follow-up test in the control condition of Ramani and Siegler's experiment. To get a comprehensive understanding of the advantage of the experimental group, we also simulate another scenario in which we train the neural modules with the same number of batches as in the pre-test training phase ("*n*-batch post-test training" simulation). This gives us a sense of how the model will continue evolving if we train it on more than one batch, although it is unlikely that participants actually got that much training during the 9-week period of time in Ramani and Siegler's experiment.

When measuring the model performance on each numerical processing task, the Visual component is used to perform the numeral identification task and the number line estimation task using randomly selected digit images not used in training, and the Language component is used to perform the counting task, starting from 0, until an error is made or the whole count list is completed. To measure the performance of the number line estimation, we report the linearity of the estimation (measured by the square of the coefficient of correlation R^2 between the models prediction and the target) and the slope of number line estimation. These are the same dependent variables as measured in (Ramani & Siegler, 2008). A perfect number line estimate should yield a R^2 equal to 1.0 and a slope equal to 1.0. For the magnitude comparison task, we randomly select images of two different numbers from 1 to 10 and feed them to the Visual component separately. We then use the output of its *magnitude* readout layer to determine a response, i.e., the one with larger magnitude output is determined to be the "bigger number". The training and the testing were simulated 20 times with different random initialization of the network parameters and random sampling of the data.

Results

Accuracies of the Cognitive Components across the Training

We find consistent patterns in results across all the tasks that were tested (numeral identification, number line estimation, counting and magnitude comparison). As expected, up to the point when number board game was introduced, accuracy did not differ between the experimental condition and the control condition were. However, after the number board game was introduced, across all tasks the experimental condition demonstrated better performance than the control condition. Figure 4 shows the learning curves of different tasks in the "*n*-batch post-test training" simulation.

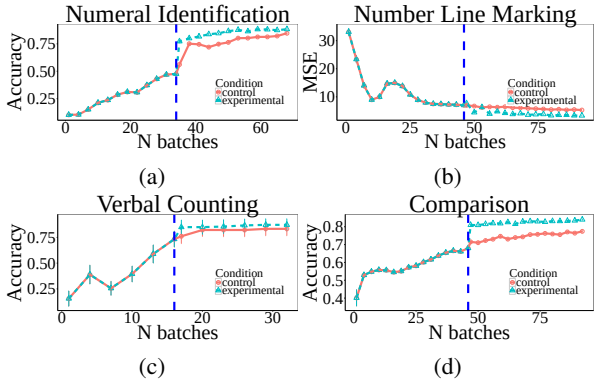


Figure 4: (a) Accuracy of the Visual component on the numeral identification task. (b) Mean Square Error of the Visual component on the number line estimation task. (c) Accuracy of the Language component on the counting task. (d) Accuracy of the magnitude comparison task. The vertical blue lines indicate the time points when number board game is introduced in the experimental condition.

Pre-test, Post-test and Follow-up Test

To compare our results with the results in Ramani and Siegler (2008), particularly their Figure 2, we also plot the pre-test scores, the post-test scores and the follow-up scores in the "1-batch post-test training" simulation. We run several paired *t*-tests to compare the pre-test scores and the post-test scores, as well as the pre-test scores and the follow-up test scores. In the control condition, as expected because

there is no training between the pre- and post-tests, the post-test scores are not significantly different than the pre-test scores. However, in the experimental condition, across all tasks the post-test scores are significantly higher than the pre-test scores (Table 1). The same results hold true for the comparison between the follow-up measures and the pre-test.

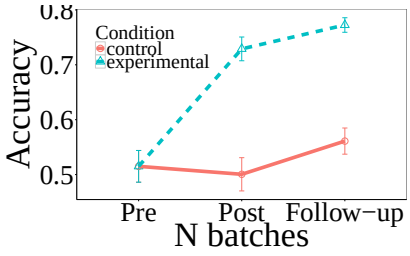


Figure 5: The model performance at the pre-test, the post-test, and the follow-up measures on the numeral identification task.

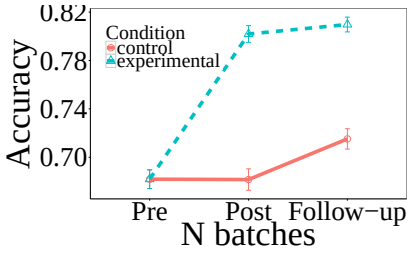


Figure 6: The model performance at the pre-test, the post-test, and the follow-up measures on the magnitude comparison task.

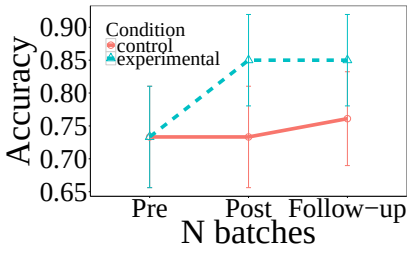


Figure 7: The performance of the Language component at the pre-test, the post-test, and the follow-up measures on the counting task.

As shown in Table 1, in the experimental condition, after the model is trained on the number board game, the numeral identification accuracy increases from 52% to 73%, the magnitude comparison accuracy increases from 68% to 80%, the counting performance increases from 73% to 85%, the square of the coefficient of correlation between the model's prediction and the target in number line estimation improves from 0.29 to 0.62, and the slope increases from 0.12 to 0.47. No improvement is observed in the control condition.

	Pre	Post	FU	Post - Pre	t_1	FU - Pre	t_2
Experimental Condition							
Numeral Identification	0.52	0.73	0.77	0.21***	5.91	0.26***	7.65
Magnitude Comparison	0.68	0.80	0.81	0.12***	12.66	0.13***	11.46
Counting	0.73	0.85	0.85	0.12*	2.25	0.12*	2.25
Number Line Linearity	0.29	0.62	0.64	0.33***	4.81	0.36***	5.99
Number Line Slope	0.12	0.47	0.46	0.35***	11.31	0.34***	13.97
Control Condition							
Numeral Identification	0.52	0.50	0.56	-0.01	-0.94	0.05*	2.41
Magnitude Comparison	0.68	0.68	0.72	0	-0.03	0.03***	3.56
Counting	0.73	0.73	0.76	0		0.03	1.75
Number Line Linearity	0.29	0.30	0.41	0.01	0.20	0.12	1.95
Number Line Slope	0.12	0.10	0.18	-0.01	-0.61	0.07*	2.99

*$p < .05$, **$p < .01$, ***$p < .001$.

Table 1: Mean scores of the different tasks at the pre-test (Pre, 1st column), the post-test (Post, 2nd column) and the follow-up test (FU, 3rd column); Differences of the scores between the post/follow-up test and the pre-test(Post-Pre/FU-Pre, 4th/6th column) and the corresponding t statistics (t_1/t_2, 5th/7th column) in the paired t-test (df=20).

Viewed from a different perspective of the data, we also show that although the performances in the two conditions do not differ at the pre-test, there is a significant difference between the two conditions at the post-test (Table 2) across all tasks. Such gap remains significant in all the follow-up measures except for the counting task.

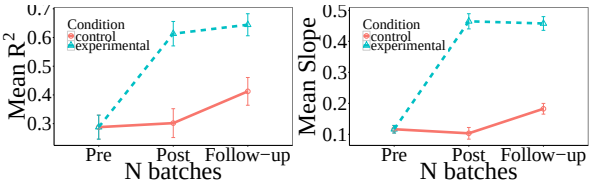
	Post	t_3	FU	t_4
Experimental - Control				
Numeral Identification	0.23***	6.06	0.21***	7.05
Magnitude Comparison	0.12***	10.09	0.09***	8.30
Counting	0.12*	2.25	0.09	1.90
Number Line Linearity	0.31***	4.34	0.23***	4.24
Number Line Slope	0.36***	11.56	0.28***	8.88

*$p < .05$, **$p < .01$, ***$p < .001$.

Table 2: Differences of scores between the experimental condition and the control condition at the post-test (Post, 1st column) and the follow-up test (FU, 3rd column) with their corresponding t statistics (the 2nd and the 4th column).

Linearity of the Number Line Estimation

One of the major motivations of Ramani and Siegler's work was to test whether playing the number board game could improve the linearity of children's number line estimation. As can be seen in Table 1 and Figure 8, in the experimental condition, both the linearity (measured by the square of the coefficient of correlation between the models prediction and the target) and the slope of number line estimation significantly increase after playing the number board game, and these learning outcomes are still significant at the follow-up test (Table 1).



(a) (b)

Figure 8: The linearity (a) and the slope (b) of the model performance in the number line estimation task evaluated at different stages.

Discussion

In the current study, we created a neural network model to provide a mechanistic account for the enhancement of numerical processing skills that was induced by playing a number board game (Ramani & Siegler, 2008; Ramani, Siegler, & Hitti, 2012). We hypothesized that various parts of the number board game actually train different components of the player that could later be used to perform other numerical tasks, such as numeral identification, magnitude comparison, counting and number line estimation. We reproduced the empirical learning effect in our computational model that implemented our hypothesis.

In our model, different cognitive components need to coordinate with each other to perform the task. For instance, as the Action component outputs the right actions, the agent constantly gets good teaching signals from the external environment which can be used by the Visual component and the Language component. As the Visual component recognizes the identities of the digits on the board, it further provides the language component with the proper inputs needed to predict the next number word. At the same time, as the agent's token moves along the board, the Visual component learns to associate the symbolic numeral with distance along the number line and the Visual and Language components jointly determine the number word to be uttered.

One limitation of the current paper is that we did not explicitly train the model's attention, e.g., we feed the Visual component with images of the current digits treating the "MOVE" action as simultaneously moving the player's token and shifting the focus of attention. We assume that the participants could allocate their attention to different parts of the environment and deploy their cognitive components in a synergistic way. Coordination of these processes to actually play the game requires executive control and working memory (Barnes et al., 2016). In future work we will explore how the selective attention and the executive control ability of the model can also be learned through training. This is a promising direction since recent advances in language grounding research in the artificial intelligence field have shown that neural network models can learn to attend different parts of an image in order to answer questions about the image (Yang, He, Gao, Deng, & Smola, 2016).

Another limitation is that we did not fully model the color board game control condition in Ramani and Siegler's work, i.e., we only modeled the aspect that participants did not receive any number-related training during the color board game playing, but we did not simulate the color-related training. In future work, we will fully model this color board game control condition. In addition to this control condition, in later studies researchers also compared the count-on procedure (reciting the number words for each new tile reached) used in Ramani and Siegler's study with the standard count-from-1 procedure (reciting the number words corresponding to the number of onward steps), and found that playing the same game using the standard procedure led to considerably less transfer to the other tasks (Laski & Siegler, 2014). This might occur because participants' attention was not directed either to the token position on the board or to the numerals on each square, as the task can be performed without this information. Also, the count-from-1 procedure provides no practice counting beyond the number two. It would be interesting to see whether modeling those control conditions within a network that learns to deploy its attention could provide a mechanistic account for why one of the interventions worked while the others did not.

In summary, the current work is a first step towards building a comprehensive computational model for numerical cognition that coordinates different modalities and integrates various training stimuli and paradigms. Our approach allows us to simulate the acquisition of number concepts as a process through which a set of component skills are assembled in different configurations in diverse task settings, promoting transfer across tasks that share components.

References

Andreas, J., Rohrbach, M., Darrell, T., & Klein, D. (2016). Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 39–48).

Barnes, M. A., Klein, A., Swank, P., Starkey, P., McCandliss, B., Flynn, K., ... Roberts, G. (2016). Effects of tutorial interventions in mathematics and attention for low-performing preschool children. *Journal of Research on Educational Effectiveness*, *9*(4), 577–606.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248–255).

Halberda, J., Mazzocco, M. M., & Feigenson, L. (2008). Individual differences in non-verbal number acuity correlate with maths achievement. *Nature*, *455*(7213), 665.

Harnad, S. (1990). The symbol grounding problem. *Physica D: Nonlinear Phenomena*, *42*(1-3), 335–346.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 770–778).

Johnson, J., Hariharan, B., van der Maaten, L., Hoffman, J., Fei-Fei, L., Zitnick, C. L., & Girshick, R. (2017). Inferring and executing programs for visual reasoning. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 3008–3017).

Lakoff, G., & Núñez, R. (2000). *Where mathematics comes from: How the embodied mind brings mathematics into being*. Basic Books.

Laski, E. V., & Siegler, R. S. (2014). Learning from number board games: You learn what you encode. *Developmental Psychology*, *50*(3), 853.

LeCun, Y., Cortes, C., & Burges, C. (2010). MNIST handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, *2*.

Libertus, M. E., Feigenson, L., & Halberda, J. (2011). Preschool acuity of the approximate number system correlates with school math ability. *Developmental Science*, *14*(6), 1292–1300.

Park, J., & Brannon, E. M. (2014). Improving arithmetic performance with number sense training: An investigation of underlying mechanism. *Cognition*, *133*(1), 188–200.

Ramani, G. B., & Siegler, R. S. (2008). Promoting broad and stable improvements in low-income childrens numerical knowledge through playing number board games. *Child Development*, *79*(2), 375–394.

Ramani, G. B., Siegler, R. S., & Hitti, A. (2012). Taking it to the classroom: Number board games as a small group learning activity. *Journal of Educational Psychology*, *104*(3), 661.

Yang, Z., He, X., Gao, J., Deng, L., & Smola, A. (2016). Stacked attention networks for image question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 21–29).