Can a Recurrent Neural Network Learn to Count Things?

Mengting Fang (mtfang@mail.bnu.edu.cn)[1], Zhenglong Zhou (zzhou34@jhu.edu)[2],
Sharon Y. Chen (syc2138@columbia.edu)[3], James L. McClelland (jlmcc@stanford.edu)[4]
[1]Department of Mathematics, Beijing Normal University, Beijing, China
[2]Department of Cognitive Science, Johns Hopkins University, Baltimore, MD, 21218
[3]Department of Computer Science, Columbia University, New York, NY, 10027
[4]Department Psychology, Stanford University, Stanford, CA, 94305

Abstract

We explore a recurrent neural network model of counting based on the differentiable recurrent attentional model of Gregor *et al.* (2015). Our results reveal that the model can learn to count the number of items in a display, pointing to each of the items in turn and producing the next item in the count sequence at each step, then saying 'done' when there are no more blobs to count. The model thus demonstrates that the ability to learn to count does not depend on special knowledge relevant to the counting task. We find that the model's ability to count depends on how well it has learned to point to each successive item in the array, underscoring the importance of coordination of the visuospatial act of pointing with the recitation of the count list. The model learns to count items in a display more quickly if it has previously learned to touch all the items in such a display correctly, capturing the relationship between touching and counting noted by Alibali and DiRusso. In such cases it achieves performance sometimes thought to result from a semantic induction of the 'cardinality principle'. Yet the errors that it makes have similarities with the patterns seen in human children's counting errors, consistent with idea that children rely on graded and somewhat variable mechanisms similar to our neural networks.

Keywords: mathematical cognition; numerical cognition; neural networks; development; learning; transfer learning.

Introduction

Until recently it was often supposed that an understanding of the natural or counting numbers is an inalienable feature of mind, but most researchers no longer hold this view. The finding that some cultures lack exact numbers and that adult members of these cultures cannot establish exact numerical correspondence (Gordon, 2004) has led to reconsideration of this position. Several works (e.g. Izard *et al.* 2008) see counting drawing on pre-requisite 'core knowledge' systems, but still see learning to count as also depending on some sort of learning process. One view is that this process involves the 'semantic induction' (Sarnecka & Carey, 2008) of a principle (the cardinal principle), which then supports the ability to carry out specific number-related tasks. Yet another perspective (Davidson, Eng and Barner, 2012) holds that induction of a principle may not be required to achieve success in counting. Support for this perspective comes from these authors' and others' findings, showing that children who can succeed at what is often called the 'how many' task or even the somewhat harder 'give N' task may fail many other tests thought to reflect what it means to understand the cardinal principle (see also Izard et al, 2008). Thus, it is possible that true understanding of the natural numbers may not be required to succeed in counting tasks, in spite of the belief (espoused by the German number theorist Kronacker) that God made the natural numbers (Weber, 1893).

The work we report here explores how a system could learn to carry out the 'how many' or (as we shall call it, the 'count the blobs' task) as a stepping-stone toward a more complete understanding of natural number. This task may be one of the first exact number tasks mastered by young children when they are learning to count. Our goal is to explore these questions:

Competence: Can a recurrent neural network architecture that can move its center of attention across a series of 'glimpses' learn to count the number of items in a display?

Development: Does this architecture allow us to capture features of the developmental progression of performance seen in children as they learn to count?

In addition to these questions, our work is guided by the intention to explore two aspects of the nature of the setting in which learning to count takes place.

Teacher guided learning. A central guiding principle of our project is the view that learning to count consists, at least in part, of a process that occurs while a learner is engaged a learning setting with a teacher. This perspective differs sharply from that of many researchers, who see mathematics learning as a self-directed discovery process. While we agree that discovery based learning is important throughout development, we would argue that quite a lot of what we learn is shaped in part by the instructional interactions we have with those around us.

Cumulative and concurrent multi-task learning Often learning to perform a task is considered in isolation, and that has often been true in neural network research. But in reality, we learn to count within a setting where we are also learning many other tasks. Our work aims to allow us to exploit obvious commonalities between tasks in the

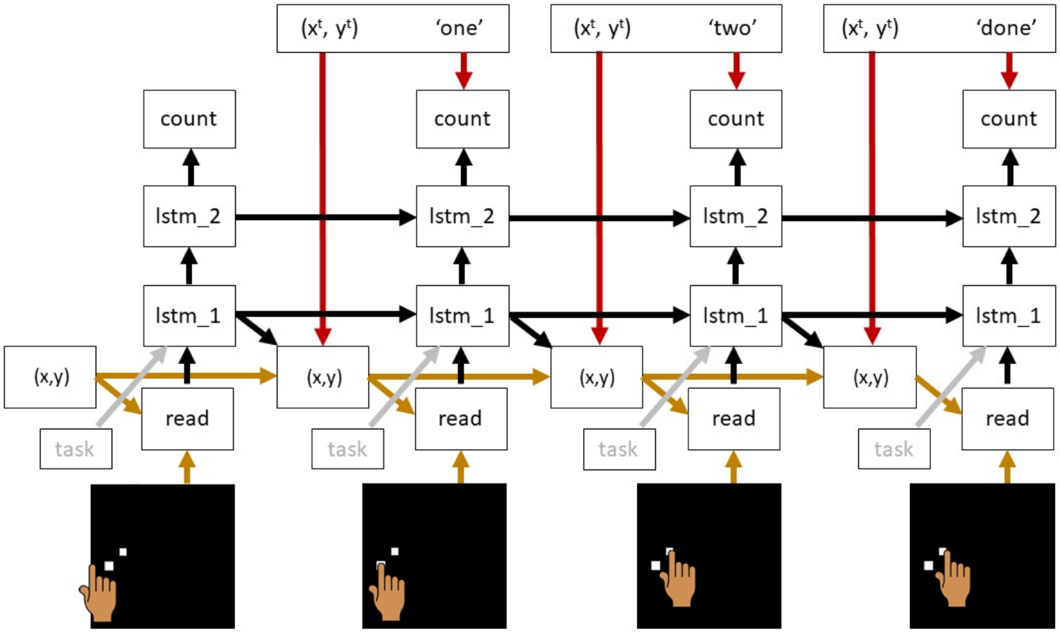

Figure 1. The differentiable recurrent attentional counting (DRAC) model, shown processing an input containing two blobs to count, along with the teaching signals specifying there to look/point and what to say when counting each blob in the display. Red arrows indicate where look/point and count-word teaching signals are injected during learning in the *count-the-blobs* task. Black arrows indicate linear mappings (each considering of a weight matrix and vector of bias weights) subject to modification during learning. Gold arrows specify unmodifiable pathways associated with managing the model's point of fixation on the input. The task box and associated grey arrow indicate how task information is provided to the network. This module is only present in the task-controlled version of the model (DRAC_{tc}). The base version of the model only learns to perform the count-the-blobs task. The task-controlled version also learns the *count-to-nine'* task and the *touch-the-blobs* task. In the count-to-nine task, the input is always blank (no blobs present), and there is no look/point teaching signal. In the touch-the-blobs task, the input is an array of blobs as in the count-the-blobs task, but no count-word target is provided.

number domain, to understand how prior or concurrent learning of one task can support learning of another task.

Description of the Focal Task

The main task we set our network might best be called the 'count the blobs' task. On each trial of the task, a display is presented containing from 1 to 15 blobs (see Figure 1) in a 50x150 pixel window. One of the network's outputs is a point position, consisting of an (x,y) coordinate pair which we treat both as the center of its gaze and as the location it is touching to in the display. (In Figure 1, this is represented by the index finger of a hand, although no indication of the point position actually appears in the display.) The network starts each trial pointing to the middle of the left edge of the display area. Its task is to point successively to each of the blobs from left to right, outputting the current count as it points to the each successive blob, until there are no more blobs. Upon reaching the last blob the network should produce an 'I'm done' or 'that's the answer' response, and should maintain its point on the last blob.

The Network and its Situation in its Environment

Our network is adapted from the DRAM architecture introduced by Gregor et al (2015). It is a recurrent neural network consisting of a 'read' module, similar to a retina, that shifts its point of focus in response to the output of a 'point' module. In our case, the network specifies the location of the next point relative to its current point of fixation, whereas DRAM specifies its next fixation point relative to an external reference point, such as the lower left hand corner of the display. We chose to specify the next point in relative terms for two reasons: First, visual targets are arrayed on the retina at positions relative to the point of fixation, and eye and movements are also specified relative to their current position. Second, it may simplify the task of learning to shift one's point to the next item to the right, an ability especially useful in counting. If one specified where to point in external coordinates, each item in an external array would be at its own unique position. If the next point is specified in relative coordinates, and if objects are laid out in a culturally conventionalized way (such as a linear array running from left to right) then the next object to count will tend to fall in

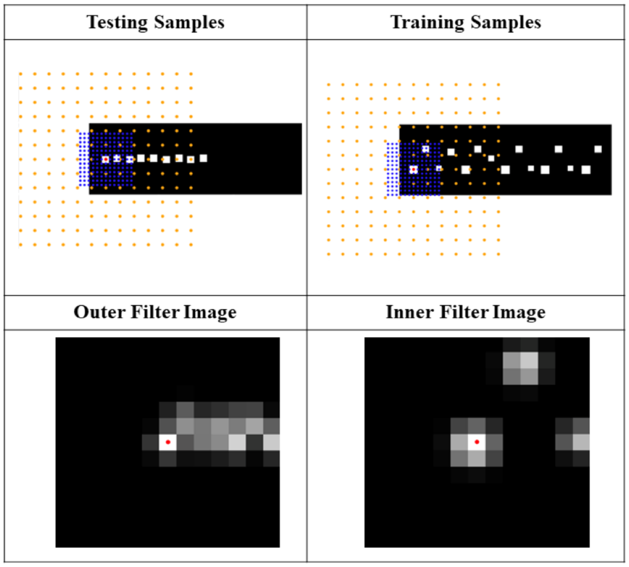

Figure 2. Examples of the stimuli used in the experiments. Top: locations of the centers of the Gaussian filters (blue: retina array; orange: periphery) while the network is fixated on the first blob in an example test display (left) and training display (right). Bottom: output of the filter arrays for the case shown in the top right panel.

a similar range of relative positions, so the stimulus for the action, and the action itself, would be very similar, when counting each successive item.

The output of the read module is a set of 2 arrays, one approximating a fovea (with relatively high resolution) and one approximating peripheral vision (with relatively low resolution) as indicated in Fig. 2. The contents of these arrays are the convolution of the input, relative to the current point of fixation, through a set of 13 x 13 evenly spaced Gaussian filters. The two arrays use a different spacing parameter s, set to 3 pixels for the fovea array and 10 pixels for the peripheral array. The centers of the filters are spaced *s* pixels apart in both the x and y directions, with the central filter centered on the current point of fixation, and the width parameter of each filter is ½ s. The positions of these filters when the point is centered on the leftmost blob in a display are superimposed on the display image; the outputs of these filters are shown in Figure 2b and c.

The 'point' module consists of an LSTM (Graves, 2013) which receives the output of the read module as well as its own prior state (initialized to all 0's at the beginning of each trial). It produces two outputs, one that is passed on to the count module and another passed through an additional layer of connection weights to produce the Δx and Δy values specifying the position of the next point, as shown in Figure 1. At the beginning of each trial, the prior state of the point module is set to all zeros, and the starting location of the point is placed in the middle of the left edge of the display area.

The network also contains a count module, whose task is to produce the appropriate count word, or the 'done' output for each glimpse. The count module also consists of an LSTM. It receives the output of the point module and its own prior state (also initialized to all 0's at the beginning of each trial), and produces a pattern of activation propagated through another layer of connection weights to an output layer consisting of 16 units, one corresponding to each of the numerals from 1 through 15 and one corresponding to the 'I'm done' output. The output of this layer is normalized using the softmax function so that the sum of the activations across these units is always equal to 1, and the response of the network is taken to correspond to the unit with the highest activation.

Training Regime

As an initial exploration of guided learning, we treat learning as occurring in two modes, both involving a student and a teacher. In the first of these modes, *teacher guided learning*, the network learner L attempts to anticipate the counting and pointing actions of a teacher T while T demonstrates how to perform the task. T's demonstration is made available to L, not act actual visual or auditory inputs, but as targets for the next count word (or 'I'm done' signal) and for the next point location. In addition, L's point of attention for the next step is forced to correspond to T's, so that the teacher's point guides where the learner is actually looking. In the other mode, *monitored performance with feedback,* L attempts to carry out the task with feedback from T. In this mode, L is thought to overtly produce counts and points, allowing T to evaluate L's performance and provide feedback (Alibali & DiRusso, 1999). In the simulations, we assume this feedback is provided in the form of a teaching signal or target for the correct count and point at each step in a counting sequence, as if T provided a correct demonstration right after L's attempt, but without forcing L's point at each step in the counting and touching sequence.

To implement these stated principles of learning, we alternate teacher guided and monitored performance trials during training. To recap, during a *teacher guided learning* trial, the network attempts to predict the teacher's count output and next point action at each step in processing the current input; the teacher's actual words and points serve as teaching signals to the learner, and its next point is forced to the correct location. During a *monitored performance with feedback* trial*,* the feedback signals are the same, but the learner's own point output, and not the teacher's, is used to determine the next point of fixation.

During training, the blobs were squares with sides of length 4, 5, or 6. The *x* position of the center of the next blob was 7 to 10 pixels to the right of the previous *x* position, and the *y* coordinate was chosen uniformly within the constraint that the whole blob remained within 10± pixels of the vertical midline of the display.

Frequency of training with arrays containing different numbers of blobs. Researchers have long observed that we encounter displays with a small number of items more frequently that displays with larger numbers of items. Accordingly, during training the frequency of a display containing N items was proportional to $1/N^2$. This means that the network encounters displays containing just 1 item 100 times more frequently than it encountered a display containing 10 items. Although we tested the network's performance on arrays containing 1 to 9 blobs (see below), the displays used in training contained up to 15 blobs. This prevented the network from learning that counting stops at a particular point within the range encompassed during testing.

Testing Regime

We evaluated the network's performance during testing sessions interspersed at different time points during learning. During each test, we assessed performance on 500 randomly generated trials of each N from 1 to 9. Test trials are like monitored performance learning trials, but without any teaching signal provided. The network simply steps through a sequence of glimpses, and the results are recorded as they would be in an experimental setting where a child's counting behavior is observed over a sequence of trials, while the experimenter provides neutral encouragement after each test item is presented. Our testing protocol followed the approach many developmental psychologists have used, making the conditions of testing as easy as possible, to give the network the greatest chance of demonstrating its mastery of the counting task (Alibali & DiRusso, 1999). Accordingly, during testing, the blobs were always 5x5 pixels, and the blob spacing was varied only by ±1 pixel in both the x and the y directions. Informal testing with less uniform sizes and positions does result in more errors, as also seen in young children (Gelman & Gallistel, 1986).

Cumulative and Concurrent Multi-Task Learning

The final element of our investigation is the exploration of how the learning of a focal task can be supported by previous and/or concurrent learning of related component tasks. Our focal, count the blobs task can be considered to involve the coordination of two simpler tasks: Touching each of the blobs in the array, and reciting items in order from the count list. The required coordination involves ensuring that there is one and only one count for each blob, so that the count stops when there are no more blobs. To explore these issues, we introduced a count to 15 task and a touch the blobs task. As in the count the blobs task, *teacher guided learning* and *monitored performance with feedback* trials alternated in both of these tasks, so that the networks' point of fixation was forced to the correct position at each step on ½ of the training trials in each task.

Count to 15 Task. This task provides the network with a way to learn the count list, without the additional demands of coordinating this process with successively touching each of the items in an array. In this task, the network was shown a blank display for 16 glimpses on each trial. The network was asked to produce a count word according to the count list from 1 to 15 for the first 15 glimpses and say 'I'm done' at the last glimpse.

Touch the Blobs Task. This task teaches the network to touch all the blobs in a canonical order, from left to right. On each trial of the task, a display is presented containing from 1 to 15 blobs with frequency decreasing with N as in the count the blobs task. The network starts each trial with its point of fixation in the middle of the left edge of the display area. Its task is to direct its focus of attention successively to each of the blobs until there are no more blobs. Upon reaching the last blob the network should maintain its point of fixation on the last blob.

Comparison of Three Learning Conditions

The results we present below come from two multi-task learning conditions, and a baseline, one task learning condition.

One-task condition (1T). In this condition, the network simply learns to perform the count the blobs task.

Three-task condition (3T). In this condition, the network receives interleaved trials on all three tasks.

Touch only then three-task condition (TF+3T). In this condition, the network learned the touch the blobs task alone until it reached a stringent performance criterion (maximum per trial point error with 10 blobs must average less than 2.5 pixels over 100 trials in each of three successive tests separated by 500 training iterations). The network then proceeded to the three-task condition, receiving interleaved training in all three tasks.

For each condition, we trained and analyzed three independent runs of the DRAC network. For each run, after network initialization, training began. In each training iteration, the network processed a single training example from each of the tasks included in the condition. After each training trial, gradients for learning were calculated, and the connection weights were updated. Testing occurred after every 500 training iterations, as described above. In the TF+3T condition, the iteration counter was reset to 0 after reaching criterion on the touch the blobs task.

Results

Competence after Learning

We assessed the counting competence at each test point. To do so, we determined whether the network produced a correct count sequence and a correct point sequence on each test trial. For a count sequence to be correct, the count response had to progress through the correct count sequence from 1 to N and then end with the 'I'm done' response. For a point sequence to be correct, the point had to fall within a window of ±3 pixels vertically and horizontally around each successive blob center; after correctly touching each blob once, the network had to touch the last blob a second time for the sequence to be scored correct. Using these measures, we

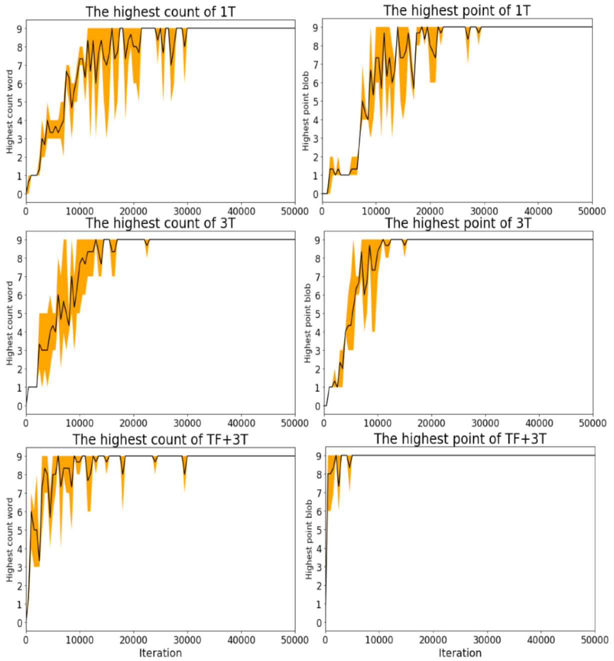

Figure 3. Left: highest count achieved at different test points. Right: Highest number of blobs touched correctly during counting.

could then define a criterion for perfect test performance on the count the blobs task. For performance to be perfect, both the count sequence and the point sequence had to be correct on *all* trials at *all* values of N – no error of either kind on any trial.

Applying the criterion above, we found that all three networks in each of the three learning conditions achieved perfect performance in well under 30,000 trials, but then made errors on some later test points. By about 30,000 trials, all networks stopped making any errors, maintaining perfect performance thereafter. It is striking that an approximate, graded, gradient-based learning system can achieve such a high standard of accuracy in this task.

Relation between Learning to Touch and Learning to Count

We next considered whether cumulative and concurrent learning can lead to cross-task transfer in our network, focusing on counting performance *per se*. Here we drew inspiration from the work of Alibali and DiRusso (1999) who found that children who were better able to correctly touch a sequence of objects also succeeded better at counting them. In Alibali and DiRusso's study, only the count was considered in determining if a child performed correctly in their counting task. We adopted a similar approach, scoring whether the network's count sequence was correct, regardless of its pointing performance, then considering the relationship between counting performance measured in this way and pointing performance. For this assessment, we measured the highest number of blobs each network could count correctly, defining 'count correctly' for a given N as achieving a score of 90% correct count sequences on the 500 test items containing N blobs. We plot the results in Figure 3 for each training condition separately, with the mean over the three runs shown as a solid black line and the range of highest count scores indicated by the yellow bands around the mean performance. We stress that all three networks had the same amount of training in counting blobs.

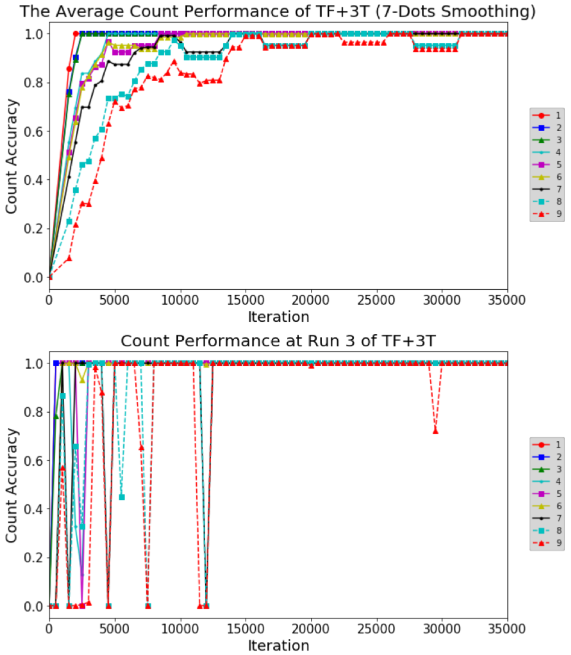

Figure 4. Count performance on displays with different numbers of blobs in the network trained to touch each blob before proceeding to the three-task training phase. Top: Results averaged across the three networks and smoothed to reduce variability. Bottom: Performance of one of the three networks, showing rapid initial learning, followed by occasional backsliding on displays containing larger numbers of items.

The figure reveals that learning to count occurred most quickly in the TF+3T condition. Average highest count for the networks in this condition reached 6 by iteration 1,000, and each of the three networks first achieved a perfect counting score after at most 5500 training examples, though there was some slippage in counting to 8 or 9 thereafter. In contrast, the first perfect count score did not occur until much later in the 3T condition or the 1T condition.

The left panels of the figure allow us to explore how the ability to count relates to the ability to point. As expected, learning to point is slowest in networks trained to do only one task; is somewhat faster in the networks trained on all three tasks concurrently; and is near-perfect from early on in networks trained to touch before three-task training. (The networks trained to touch first show a transient disruption at the start of the three task phase, then recovers to near-perfect performance very quickly.) In summary, more experience, and earlier ability, in pointing are associated with mastering the count the blobs task more quickly.

Transitions in Performance and Error Patterns

Two final issues we consider are the developmental course of mastering the ability to count larger and larger numbers of things, and the nature of the network's counting errors. Several authors (e.g., Le Corre, Van de Walle, Brannon, and Carey, 2006) have argued for a categorical divide, separating children into those who can succeed at counting only very small numbers of things (up to 3 or possibly 4) and those who can count things up to the limit of the length of their count list, called 'cardinal principle' or CP knowers. Given this, we wondered whether our models might divide into the same two categories, and whether, at some point in development, individual networks would make sudden transitions from the former to the latter group.

We were able to explore this in the data from the networks in the touch first then 3-task condition. The networks learn the count-to-fifteen task, corresponding to reciting the count list within the first 1,000 training iterations, and all already know how to point to all the blobs before they begin to learn to count them. In the top panel of Figure 4, we show the results indicating the average performance in counting blobs for these networks, smoothed to make the trends in learning different values of N more apparent. Very quickly, these networks master counting up to 3, then shortly thereafter, they master the ability to count to 4, 5, and 6, with 7 lagging only slightly behind. For comparison, in Le Corre *et al,* a child is counted as a CP knower if they can give 6 objects correctly 2/3 of the time. While counting N things is easier than giving N things, the networks in the top panel of Figure 4 would meet this criterion after only 2000 training trials. Prior to this, they would have scored as subset knowers by this criterion. It is also noteworthy that children make more errors when counting larger arrays (Alibali & DiRusso, 1999), similar to our networks.

We also scored the errors made in counting from 1 to 9 items in the same networks. After the first test point when each network counted perfectly (3500 iterations in the network shown in the bottom panel of Figure 4). All of the errors were well-formed counts of either one or two more or one less than the correct number of items. For example, the errors made by the network shown in the bottom panel of Figure 4 were always over- or under-counts of exactly 1.

Discussion

We began our exploration of the DRAC network with a question: Can a recurrent neural network learn to count things? Our findings clearly support a yes answer to this question. In all three of the learning conditions, all networks mastered the ability to reliably proceed through all of the items in an array, touching each one in turn while producing the next number word in the count list. These findings support the view that learning to count may not require a special mechanism designed to acquire the counting principles; instead it may depend on the reliance of a powerful, yet general purpose learning system, coupled with an environment that provides teacher-guided learning activities, like the ones we provide for our network.

Our findings also demonstrate a great deal of cross-influence between the ability to count and touch all the blobs in a display. In the networks that had already master the touch-the-blobs task, the ability to count up to 6 blobs correctly could be acquired in only about 2,000 training trials counting displays containing from 1 to 15 items.

The question of whether we should believe that learning to count involves a semantic induction has been a subject of controversy in the literature. While some have argued that it is (Le Corre *et al,* 2006; Sarneca & Carey, 2008), others have argued against this position (Davidson *et al,* 2012), based on the finding that performance in a wide range of tasks exhibits graded, rather than all-or-nothing acquisition. The pattern of errors in Alibali and DiRusso (1999) likewise attests to counting as a graded ability, one that varies in its success with the degree of support provided by allowing children to point to, or better yet to actually touch, the objects they are counting, and with size of the set of objects to be counted. Our current effort will certainly not fully resolve this issue, but we hope it will encourage further exploration of approaches that attempt to capture both the achievements and the limitations of human counting, viewed as a learned skill acquired gradually in a supportive learning environment.

References

Alibali, M. W., & DiRusso, A. A. (1999). The function of gesture in learning to count: More than keeping track. *Cognitive development, 14*, 37-56.

Davidson, K., Eng, K. & Barner, D. (2012). Does learning to count involve a semantic induction? *Cognition, 123,* 162-173.

Gelman, R., & Gallistel, C. R. (1986). *The child's understanding of number*. Harvard University Press.

Gordon, P. (2004). Numerical cognition without words: Evidence from Amazonia. *Science, 306(5695),* 496-499.

Graves, A. (2013). Generating sequences with recurrent neural networks. *arXiv preprint*. arXiv:1308.0850.

Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., & Wierstra, D. (2015). DRAW: A recurrent neural network for image generation. *arXiv preprint* arXiv:1502.04623.

Izard, V., & Dehaene, S. (2008). Calibrating the mental number line. *Cognition, 106(3),* 1221-1247.

Le Corre, M., Van de Walle, G., Brannon, E. M., & Carey, S. (2006). Re-visiting the competence/performance debate in the acquisition of the counting principles. *Cognitive psychology, 52(2)*, 130-169.S

Sarnecka, B. W., & Carey, S. (2008). How counting represents number: What children must learn and when they learn it. *Cognition, 108(3),* 662-674.

Weber, H. (1893), Leopold Kronecker. *Mathematische Annalen,* **43**: 1–25, doi:10.1007/BF01446613.