A Computational Model of Learning to Count in a Multimodal, Interactive Environment

Silvester Sabathiel (silvester.sabathiel@ntnu.no)
Department of Computer Science and Department of Teacher Education
NTNU – Norwegian University of Science and Technology
7491 Trondheim, Norway

James L. McClelland (jlmcc@stanford.edu)
Department of Psychology, Stanford University
Stanford, CA, 94305 USA

Trygve Solstad (trygve.solstad@ntnu.no)
Department of Teacher Education, Faculty of Social and Educational Sciences
NTNU – Norwegian University of Science and Technology
7491 Trondheim, Norway

Abstract

When learning to count, children actively engage with a variety of counting tasks and observe demonstrations by more knowledgeable others. We investigate how a single neural network-based agent, situated in a multimodal learning environment, can learn from observing such demonstrations to perform multiple number tasks such as counting temporally and spatially distributed objects, and a variant of the give-N task. We find that i. the agent can learn different tasks that require counting, ii. learning progresses in similar stages for different tasks, iii. sequential learning of subtasks aids learning of the full task of counting spatially distributed objects, and iv. a mechanism for updating memory when each object is counted emerges from learning the task. The work relies on generic deep learning processes in widely used neural network modules rather than mechanisms specialized for mathematics learning, and provides an architecture in which aspects of a sense of number emerge from learning several different number related tasks.

Keywords: mathematical cognition; neural networks; learning to count; situated multimodal learning.

Introduction

What cognitive mechanisms underlie the development of the uniquely human concept of natural numbers? Determining the exact quantity of a set of objects depends on the procedure of counting, arguably a child's first socially acquired mathematical skill. The learning process involves the coordination of two central aspects of natural numbers: *ordinality*, the position of an object relative to other objects in an ordered set, and *cardinality*, the number of objects in the set (Brainerd, 1973). Although the cognitive mechanisms underlying this coordination is a topic of ongoing debate (Davidson, Eng, & Barner, 2012; Alcock et al., 2016), at least three stages of the conceptual development of natural numbers have been characterized (Gelman & Gallistel, 1978; Clements & Sarama, 2009): 1) Reciter stage, where children learn to say the correct sequence of number words, 2) Corresponder stage, where children start to set the number words in one-to-one correspondence with different kinds of entities, such as objects or events and 3) Cardinal principle (CP) knower stage, where children relate the last ordinal number of the counted set to the cardinality of the set for any given number in their number list. The CP knower stage is often determined using the give-N task, where children are asked to give exactly N objects to a puppet or target position. However, it has recently been argued that children can master this task in an instrumental manner without a full understanding of the concepts of cardinality or natural numbers, and that the complexity of learning these concepts has been underestimated (Davidson et al., 2012).

Children with a fully developed concept of the natural numbers are expected to generalize the counting procedure to a) counting several kinds of objects and entities, like temporal events of verbal or motor units (abstraction), b) counting a set of objects in any order (order irrelevance), and c) an arbitrarily long list of number words (Gelman & Gallistel, 1978; Clements & Sarama, 2009). It is now increasingly accepted that the abstract concepts of counting and natural numbers involve the integration of multiple cognitive and visual-motor skills, and these concepts are learned through active exploration of and interaction with the child's environment. These explorations and interactions involve multimodal experiences and activities, and include instruction and feedback from more knowledgeable others (Anghileri, 2000).

One approach to gaining a deeper understanding of the conceptual development of natural numbers is to develop and investigate computational models trained on tasks similar to those that children perform. So far, computational models of the cognitive processes underlying our concept of number have predominantly focused on a single counting task or aspect of the natural numbers in highly abstracted contexts. Examples include the investigation of how recurrent neural networks can learn to count temporally distributed features encoded as a binary string (Wiles & Elman, 1995), and how a computational system can generalize from a finite list of number words to a recursive number concept (Piantadosi, Tenenbaum, & Goodman, 2012). We take inspiration from the idea that human cognition is rooted in the perceptual and physical interactions of the learner and others carrying out relevant tasks in a shared external environment (Tran, Smith, & Buschkuehl, 2017). In this setting, an agent can learn by predicting the actions and words of others demonstrating how

to perform learning-relevant tasks such as counting the objects in a display. Fang, Zhou, Chen, and McClelland (2018) demonstrated that a recurrent neural network can learn to count squares arranged in a linear array in this way. Though a step into exploring the acquisition of number, their network could be said to have simply acquired a single procedure for counting objects, rather than an understanding of numbers.

It has been suggested that a sense of number emerges, not from a moment of insight associated with a principle that characterizes the meaning of number, but from learning to carry out a range of different number related tasks (Davidson et al., 2012). We take inspiration from this idea here, investigating how a neural network model can learn to perform several different number-related tasks, sharing the knowledge in its connections across the tasks. We support the view that the agent is acquiring aspects of an overall sense of number by showing that learning in some of the tasks is facilitated by prior learning of some of the other tasks.

Set-up and Methods

Learning Environment

The virtual learning environment provides an artificial agent with a multimodal, interactive interface. The agent can receive commands, express a limited set of words, and has a simplified, virtual 'hand' that it can move around and use to touch, pick up, and release objects. The world is a 4x4 grid with two binary features at each grid point, one signaling the presence of an object at the grid location and another signaling the presence of the agent's hand (Fig. 1). The environment allows a set of motor and linguistic outputs to the world and allows teaching signals corresponding to these outputs to guide learning. The set of motor actions are one-step movements of the hand in 2D space (left,right,up,down), as well as touch, pick up and release actions. The language output consists of the count words 'one' to 'nine', and the word *'Stop'*.

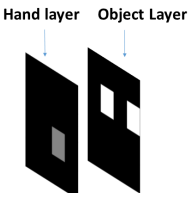


Fig. 1: 2-layered visual input - the gray square represents the hand and the white squares represent the objects to be counted in the environment.

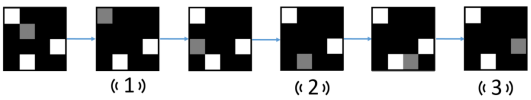
Task descriptions

The agent was trained on four different tasks commonly used to investigate number comprehension in children described in Fig. 2.

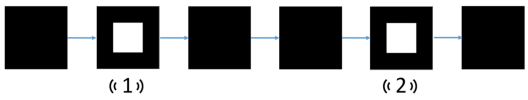
Learning system

Demonstration-driven learning We simulate a learning situation in which the agent observes and predicts the actions and words of a teacher performing the counting tasks. For each time step, the agent receives input specifying the correct action of the teacher, and adjusts its connection weights based on its prediction error using backpropagation. The approach contrasts with reinforcement learning based approaches in which the environment only sets tasks and provides rewards. We argue that this approach captures important features of the environment in which children learn, allows the model to adopt culturally-defined counting habits, and accords with evidence that children's number learning is influenced by adults' use of number to refer to items present in the child's environment (Gunderson & Levine, 2011). In the subsequent testing situation, the agent acts autonomously without feedback, essentially producing motor and language actions it learned by observing the teacher.

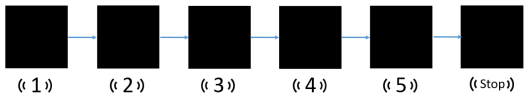
Teaching signal For the count-all-objects task there are multiple possibilities for which order to count the squares and which trajectory to take between them. The solving algorithm which provides the teaching signal for our agent always chooses to count the closest square to its position (measured by the euclidean distance). If there are several squares equally



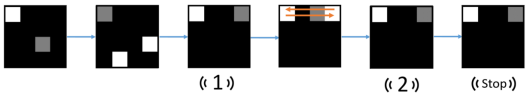
(a) *Count all objects:* The agent is to touch each object exactly once and say the corresponding number word as each object is touched.



(b) *Count all events:* The agent is to say the correct number word after each flash of a white square in the visual input.



(c) *Recite-N:* The agent is given an arbitrary number, N, from 1 to 9 as input in the language channel and is to recite the number words from 1-N and say *'stop'* afterwards.



(d) *Give-N*: The agent is given an arbitrary number, N, from 1 to 9 as input in the language channel and is to move N white squares from an infinite source of white squares in the left upper corner of the environment to the target in the right upper corner of the environment. The agent is to say the corresponding number word after each given square and say *'stop'* after N objects have been moved.

Fig. 2: Description of the four counting tasks and illustration of a solution process including the visual output and verbal input. Spoken words are denoted below the image. White squares represent manipulable objects and the grey square represents the movable hand. Orange arrows denote the movement of the hand between time instances.

far away, one of them is chosen randomly. The trajectory of the hand to the aimed object is determined by the relative coordinates between the hand and the aimed square at each time step. If the horizontal distance between them is larger than the vertical one, the manipulator will move in the horizontal direction and vice versa. If the horizontal and vertical distance between them is equal, horizontal movement is prioritized. The same applies for the trajectories of the give-N task.

Importantly, as discussed below, the scoring of performance during assessment is based on adherence to number principles, rather than the specific actions specified by the demonstration algorithm.

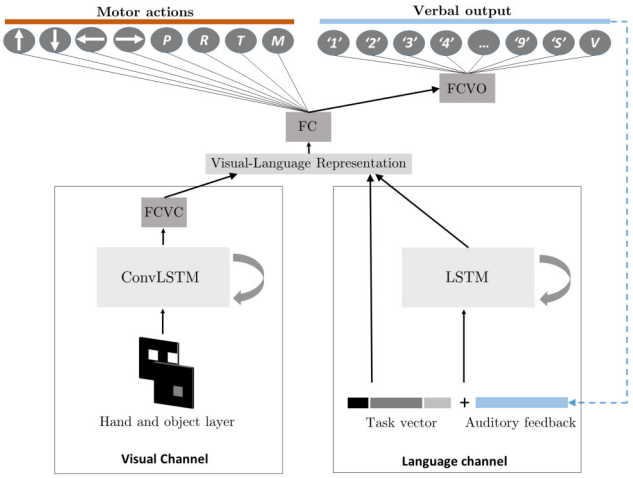


Fig. 3: Neural Network architecture with multimodal channels: The output of the visual channel (ConvLSTM) on the left side and Language Channel (LSTM) on the right side are concatenated to the multimodal 'Visual-Language' representation, from which the policy for the next motor action is computed via a fully connected layer (FC); the verbal output is computed via an additional hidden layer with ReLU and another FC layer (FCVO). Symbols on the output nodes indicate alternative actions or spoken responses. For the motor actions, arrows nodes denote the direction of the hand movement, *P* picking up, *R* releasing and *T* touching the object of the current position of the hand. *S* represents the word *Stop* and the numbers *1-9* the corresponding number words. An action is executed if the activation of node M exceeds .5, and a verbal response is produced if the activation of node V exceeds .5.

Neural Network Architecture The network architecture is illustrated in Fig. 3. The choice of architecture was motivated by the required computational capacity to process the complexity of the multimodal sensory input, action space, and temporal dependencies inherent in the counting tasks. The network architecture includes two channels, representing a visual modality and an auditory language modality:

Visual channel: The visual channel is a ConvLSTM (Xingjian et al., 2015), which takes the 4x4 image as input and is designed to have the capacity to remember long term dependencies of the visual input. The ConvLSTM is a version of an LSTM whose internal structure uses replicated channels that tile the input space, as in a feed-forward Convolutional Neural Network (CNN), allowing the network to develop a spatially structured working memory. In our architecture the CNN consists of 5 kernels of size 3x3 applied with a stride of 1 and a zero padding of size 2. A fully connected layer at the top of the visual channel (FCVC) and a ReLU-activation function maps the 2D hidden state of the ConvLSTM to a 1D vector with 40 units, consituting the output of the Visual Channel.

Language channel: The language channel receives the task instructions encoded in a layer with a total of 20 binary units (Fig. 4), where the first 5 units encode the verb (action) of the task instruction, the subsequent 10 units encode the quantifier (*1, 2, . . . , 9, ALL*) and the last 5 units encode the entity that is to be counted. Each of these words is one-hot encoded. To allow for the possibility to extend the instruction vocabulary that can be tested with the same architecture, some of the units in the task instruction have been left without meaning (and thus are not shown in Fig. 4). The language channel also receives the network's full output vector from the last time step, serving as an 'efference copy' of the linguistic activations in the network's output layer, independently of whether an overt action was emitted to the environment. These two vectors are concatenated to form the input to the LSTM. The data is then processed via a standard LSTM with an internal vector size of 11 units, with fully connected weights to learn the LSTM's gates (Hochreiter & Schmidhuber, 1997). The output of the LSTM is also an 11-unit vector.

Output: An intermediary, multimodal 'Visual-Language' representation with 71 units is produced by concatenating three vectors: The output from the visual channel, the output from the language channel, and the task instruction vector. This vector is mapped to the motor action units through a fully connected layer. The Visual-Language layer is mapped to the verbal output units via a fully connected layer with ReLU activation function to a hidden layer of 71 units, then another fully-connected feed-forward layer. All output units can take values from 0 to 1 independently (sigmoid activation function). The most active action is produced if the activation of the M unit exceeds 0.5, and the most active verbal output unit is produced if the activation of the V unit exceeds 0.5.

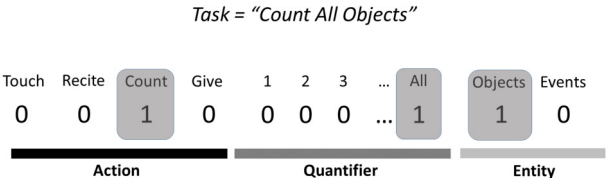


Fig. 4: Concatenated task vector, which is given as input for the language channel. The vector corresponds to a statement consisting of an action, a quantifier and an entity and is used to instruct the agent about what task to perform.

Learning algorithm The network is trained via supervised learning with an automatic solving algorithm used to create the teaching signal. For each time step, the agent predicts the action of the teaching signal, where the teaching signal decides which action is executed and therefore the next state of the environment. After each whole trial, the weights of the neural network are updated using backpropagation to minimize the sum across items in a batch and steps of the action sequence of the mean-square-error between the output vector of the network and the encoded vector for the action from the teaching signal. This learning measure is independent of the actual action which the learning system would take and thus reflects learning to imitate the actions of the teacher rather than from feedback on the correctness of its own actions. The batch size is 8 times the number of tasks to be learned in the current learning schedule. An epoch of training is defined as one full forward and backward pass of the whole batch. Instead of creating a training and test data set that cover all combinations of the multiple task parameters, the agent was trained in an 'online' fashion (Ruciński, Cangelosi, & Belpaeme, 2012), where after each trial a new batch is uniformly drawn from the tasks and the number of entities to be counted. Moreover, the actual locations of the objects, the time steps between events and the initial position of the hand are randomly chosen for each trial. For simulations of transfer learning in Fig. 7, the network was trained with the Adam-optimizer (Kingma & Ba, 2014) and a learning rate of 10^{-2} in early learning stages (until the average loss dropped to 0.2) and 10^{-3} for the upcoming epochs in later learning stages. For all other results presented in this article the network was trained using stochastic-gradient-descent (SGD) as optimizer and a learning rate of 2×10^{-1} for the early and 5×10^{-3} for later learning stages.

Assessment of performance

In order to assess the development of the networks performance on the individual counting tasks described above, network performance was tested every 200 epochs during training. At each test point the success rate over 20 test trials was measured for each number from 1 through 9. Depending on the given task a test trial was considered successful if it fulfilled the corresponding criteria, such as 1-1 correspondence, the correct number sequence and having said *Stop* after the correct number of counted or given entities.

To test whether the observed effects in the training of the neural network are statistically significant, the data for the recorded performance and the transfer effects reported in the results section are given as the average over 5 independent training runs. For each run, the neural network received random initial weights and unique random training examples.

To aid in assessing the network's stages of learning, the variability of the action sequence was measured via the normalized quantity $1 - \sqrt{\sum_i f_i}$, where f_i denotes the frequency of the *i-th* action relative to the length of the total action sequence (Allaj, 2018).

Results

Performance on each counting task

When trained separately on each task, 5 of 5 networks reached 100% accuracy for all values of N=1 to 9 in both the count-all-events and the give-N task. For recite-N 1 of 5 networks did not master N=1 perfectly, and for count-all-objects, 1 of 5 did not perform perfectly with N=2 to 9. Results from the runs where mastery did occur are reported in Fig. 5. The easiest tasks to learn, measured by the number of training epochs to reach perfect performance averaged over all numbers and runs, were the give-N task ($M = 1692, SD = 573$) and the recite-N task ($M = 1808, SD = 775$). In the literature on human learning, being able to give N items usually emerges later than counting the number of items in a display. In our simulations, the give-N task is likely easier to learn because it put lower demand on visual processing than the count-all-objects task.

The count-all-events task was the least likely to reach perfect performance, and required somewhat more epochs to learn when it did succeed. The count-all-objects task is the most complex, since it requires not only visual processing, but also coordinated action to touch all objects, and developing 1-1 correspondence between touching objects and reciting number words. Consequently, this task required the most training epochs ($M = 6215, SD = 3201$) to reach perfect performance.

To see whether a single network has the capacity to learn all tasks, we trained the model on all four counting tasks simultaneously using interleaved learning. Performance reached near perfect levels for all tasks except the count-all-objects task, which only reached 80% success rate for counting one object, and showed declining performance for increasing numbers.

The results reported above as well as those reported below were obtained with the parameters described in *Methods*. In work to be reported elsewhere, we have subsequently achieved reliable learning of all four single tasks as well as reliable learning of all tasks simultaneously. To achieve this, we added a drop-out rate of 40% to the 'Visual-Language representation' layer (excluding the task vector), increased its number of units from a total of 71 to 93, omitted the last hidden layer before the verbal output and trained the network with the same optimizer and learning rates as described in *Methods* for the transfer learning.

Stages of skill development

Although the counting tasks are all different, the underlying abstract counting principle to be learned remains the same. We asked if there are similarities in what the network learns across the separate tasks. For the three tasks requiring the establishment of 1-1 correspondence between number words and objects or events (count-all-objects, give-N, and count-all-events), we observed a common stage-wise progression of skill acquisition. To illustrate this common structure in the learning development for different tasks, we recorded the

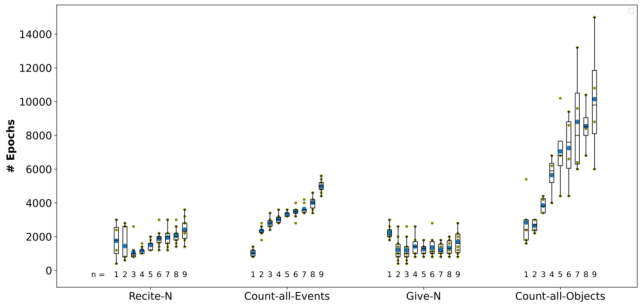


Fig. 5: Number of training epochs before the network successfully solved the task for each number in 5 different runs. Each data point is shown as grey dots. Box plots show quartiles and mean as blue dot.

success rate for the corresponding sub-skills during training. First, the network learned to respond with the same action for any input. For instance, it might respond with *7* for every time step in the count-events task. In Fig. 6 this can be seen as an initial drop in action variability as the agent adopts a stereotypical response pattern for all numbers (green lines). Second, the network increased its response variability while gradually acquiring one-to-one correspondence between number words and the appearance or touching of the entities to be counted (blue lines in Fig. 6), but typically used the same number word for all entities. Third, one number at a time, the agent learned to produce number words in the correct order (red lines in Fig. 6). The emergence of the number words in the correct order likely results from the fact that smaller numbers must be traversed when counting to larger numbers, so the network receives more practice with the smaller numbers, even though it is assessed on its ability to count to each of the numbers equally often.

Sequential learning and transfer

Children typically learn the number-word sequence before other counting-related tasks. We asked how a similar sequencing of the counting subtasks would affect learning in the artificial agent, and compared the skill acquisition of a network trained on the count-all-objects task from scratch to a network that was pretrained either a) on the recite-N task or b) to touch all objects exactly once, and subsequently trained on the count-all-objects task interleaved with the other subtask.

The comparison showed that pretraining significantly decreased the training time to perfect performance in the count-all-objects task (Fig. 7 and Table 1). The effect of pretraining the network to touch all objects was particularly strong (Fig. 7, center panel), likely reflecting that touching all objects places similar demands on the network as counting all objects. One such demand is remembering which objects have already been counted - a function we look at in more detail in

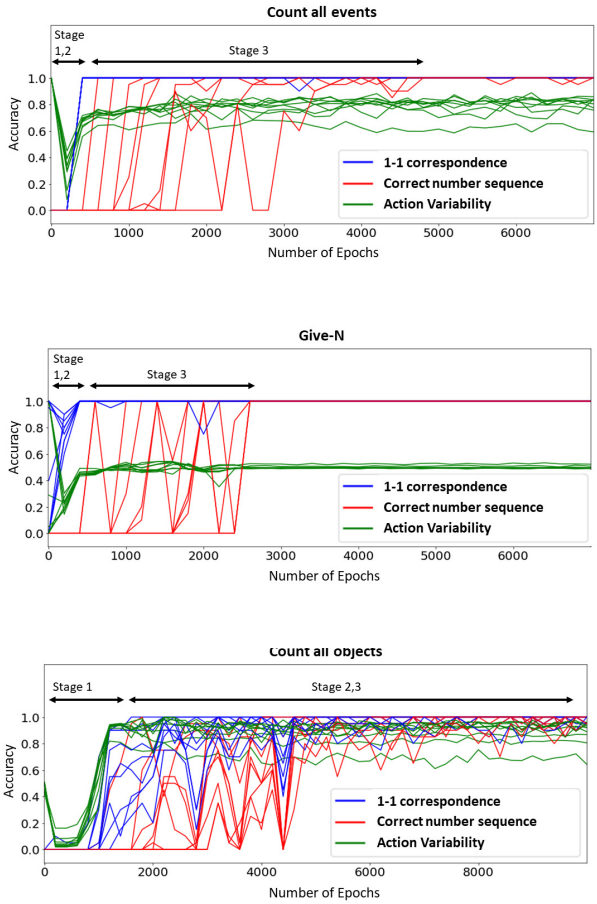


Fig. 6: Illustrations of learning stages for individual simulations of the count-all-events (top), give-N (center) and count-all-objects (bottom) tasks. Each line of the same color corresponds to a separate number of entities given in the task vector for each task.

the next section. A similar transfer effect was also observed for the count-all-events task after pretraining the network on the recite-N task.

Emergence of a memory mechanism

A major cognitive challenge during the development of counting is achieving 1-1 correspondence between the number words and the counted entities (Alibali & DiRusso, 1999). Among other things, this task requires the learner to count each item exactly once. Employing a fixed pattern of proceeding through the tiles in the array (e.g. proceeding across the top row, then down and back across the second row, etc.) would solve this subtask, but would not be efficient, especially when there is only a small number of items to count. Another way to meet this challenge is to mentally mark elements in an internal representation as elements are counted. In our setting this may be more demanding than the cognitive 'set difference' operation proposed in (Piantadosi et al., 2012), where items are simply removed from an ordered

Pretrained task	# of epochs to learn to count all objects (M±STD)
From Scratch	5390 ± 1730
Recite-N	1963 ± 1089 ($t = 9.9, p < 10^{-10}$)
Touch-all-objects	1207 ± 1101 ($t = 13.5, p < 10^{-10}$)

Table 1: Number of epochs, averaged over 5 runs and all the numbers 1-9, required to learn to count all objects for different conditions of pretraining the neural network. The t- and p-values given with respect to the mean values for the simulations, where the model has been trained from scratch.

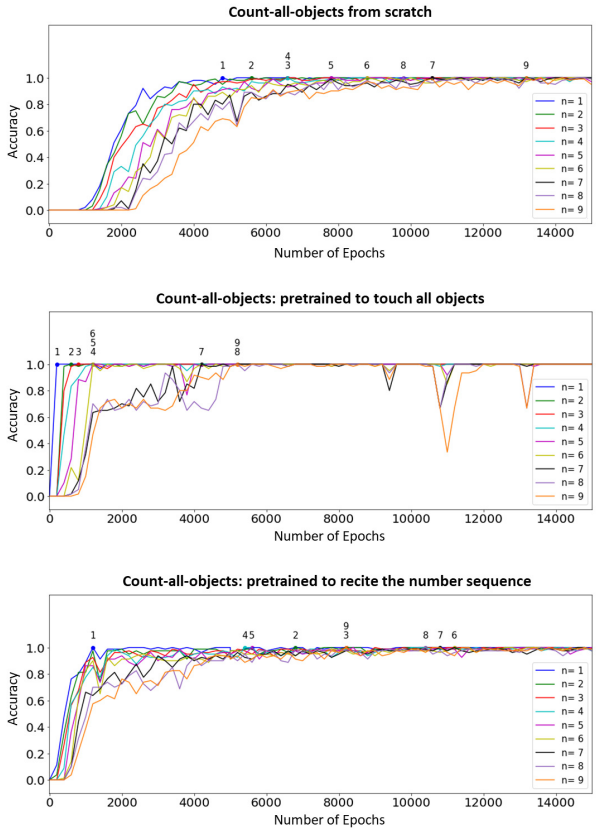


Fig. 7: Comparison of the learning development for the 'Count all objects'-task when the task is learned from scratch (top), when the computational system is pretrained to touch all objects exactly once (center) and when it has been pretrained to recite the number words (bottom). Each line shows the average over 5 simulations. The numbers on top of the graphs denote the number of training epochs it took for all 5 simulations to achieve perfect performance for that particular number.

list, because in our case the positions of the objects already counted must be considered to avoid counting them again. In order to understand how the neural network keeps track of objects it has already counted, we plotted the internal state of all layers of the ConvLSTM as the trained agent solved the count-all-objects task. We observed that one of the five internal long term memory cells acts like a memory of counted object positions: A the beginning of the task, all units of the memory cell are turned off. The unit corresponding to the position of the agent turns on, but turns off again as the agent moves. Each time the agent counts an object, the unit at that object's position stays on for the rest of the simulation (Fig. 8). A comparison of the visual input and the memory cell state will provide information about where uncounted objects are, and if all objects have been counted. Intriguingly, this implementation of a memory operation is not shown to the network; instead it emerges as it learns through anticipating the demonstrated action sequences.

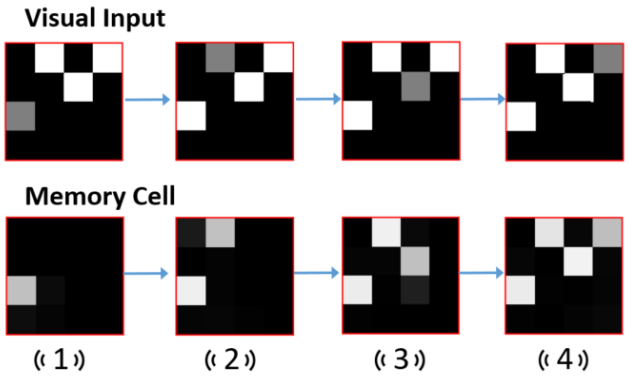


Fig. 8: Visual input and the activity of one of the five internal memory cells in the ConvLSTM when the trained system counts objects. Each time the agent counts an object, the corresponding unit in the memory cell turns on and stays on for the rest of the simulation. Symbols and colors as in Fig.2.

Discussion

A fully generalized concept of counting involves applying the counting procedure in a diverse range of settings. We investigated how a neural network model composed of standard architectural components can learn several counting-related tasks. Learning progressed in similar stages for tasks with distinct properties, like whether the entities to be counted were temporally or spatially distributed, and whether the agent had to move around to count objects or not. The common pattern of skill development may suggest that the network learned abstract features of counting that are common to all counting tasks.

This interpretation is consistent with the observation that the time required to learn a task like counting all objects was substantially reduced if the network was first trained to recite the number list or touch all objects. At the same time, we found that it was hard using initial parameters for the network to learn all tasks simultaneously, and when it succeeded in doing after refinement, successful learning of all tasks in the same network required extra training time. Future work will seek to understand this observation further. One possibility is that it reflects a case where a cumulative approach, building

from simpler component skills, is a better way to promote overall mastery.

In the count all objects task, the network learned an internal representation that could be updated as objects were touched and counted. The network developed a dynamic representation of the objects from which the objects were "mentally tagged" as they were touched and counted. The learning of such task-specific constituent operations contrasts with the idea of drawing on a hardwired general-purpose mechanism such as 'working memory' when solving cognitive tasks and may be part of the explanation for why learning genuinely new concepts is so effortful and time consuming.

An understanding of natural numbers and the counting procedure may emerge from the gradual acquisition and integration of different number-related skills (Davidson et al., 2012). The model we investigated reused previously learned number-related representations when it was trained in a sequential manner on a new task. However, the tasks it can perform do not encompass all of the number-related tasks that contribute to an understanding of number. There is clearly more to capture. Our initial progress inspires us to envision extending the approach to other tasks considered crucial to an understanding of natural numbers.

A natural extension would be to numbers larger than nine. For larger numbers, children learn to generate number words according to the place value system (Clements & Sarama, 2014). Mastery of this system takes several years of practice in an educational setting and is closely tied to the process and concept of organizing objects into groups of ten (Van de Walle, Karp, & Bay-Williams, 2013). Integrating the counting procedure for larger numbers with a corresponding grouping of physical objects into sets of ten would be a milestone for the computational modeling of the cognitive development of number and the place value concept.

Number estimation may be a potential source of intuition about number and number can also be related to positions, distances, and magnitudes. Incorporating numerosity estimation and number-position coordination tasks, as well as more explicit discourse about the links between the last number reached when counting a set of items and the number of items in the set as determined in other ways, could contribute further to the emergence of an integrated understanding of number. For this, a more fully developed language input-and-output system (based on immersion in a rich natural language context) would likely be an essential ingredient. It is also likely that exploring these issues will reveal other essential ingredients that inform the concept of natural numbers.

Acknowledgements

This work was supported by the Research Council of Norway

References

Alcock, L., Ansari, D., Batchelor, S., Bisson, M.-J., De Smedt, B., Gilmore, C., ... others (2016). Challenges in mathematical cognition: A collaboratively-derived research agenda. *J. of Numerical Cognition*, *2*(1), 20–41.

Alibali, M. W., & DiRusso, A. A. (1999). The function of gesture in learning to count: More than keeping track. *Cognitive development*, *14*(1).

Allaj, E. (2018). Two simple measures of variability for categorical data. *J. of Applied Statistics*, *45*(8), 1497–1516.

Anghileri, J. (2000). *Teaching number sense*. A&C Black.

Brainerd, C. J. (1973). The origins of number concepts. *Scientific American*, *228*(3), 101–109.

Clements, D. H., & Sarama, J. (2009). Learning trajectories in early mathematics–sequences of acquisition and teaching. *Encyc. of language and literacy development*, *7*, 1–6.

Clements, D. H., & Sarama, J. (2014). *Learning and teaching early math: The learning trajectories approach*. Routledge, UK.

Davidson, K., Eng, K., & Barner, D. (2012). Does learning to count involve a semantic induction? *Cognition*, *123*(1).

Fang, M., Zhou, Z., Chen, S., & McClelland, J. (2018). Can a recurrent neural network learn to count things? In *Cogsci.*

Gelman, R., & Gallistel, C. (1978). The childs concept of number. *Cambridge, MA: Harvard*.

Gunderson, E. A., & Levine, S. C. (2011). Some types of parent number talk count more than others: relations between parents input and childrens cardinal-number knowledge. *Developmental science*, *14*(5), 1021–1032.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Piantadosi, S. T., Tenenbaum, J. B., & Goodman, N. D. (2012). Bootstrapping in a language of thought: A formal model of conceptual change in number word learning. *Elsevier*.

Ruciński, M., Cangelosi, A., & Belpaeme, T. (2012). Robotic model of the contribution of gesture to learning to count. *2012 IEEE International Conference on Development and Learning and Epigenetic Robotics, ICDL 2012*, 1–6. doi: 10.1109/DevLrn.2012.6400579

Tran, C., Smith, B., & Buschkuehl, M. (2017). Support of mathematical thinking through embodied cognition: Nondigital and digital approaches. *Cognitive Research: Principles and Implications*, *2*(1), 16.

Van de Walle, J. A., Karp, K. S., & Bay-Williams, J. M. (2013). *Elementary and middle school mathematics: teaching developmentally*. Pearson Education: NJ, USA.

Wiles, J., & Elman, J. (1995). Learning to count without a counter: A case study of dynamics and activation landscapes in recurrent networks. In *Proceedings of the 17th annual conference of the cognitive science society* (p. 487).

Xingjian, S., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., & Woo, W.-c. (2015). Convolutional LSTM network. In *Advances in neural information processing systems* (pp. 802–810).