Simulating Problem Difficulty in Arithmetic Cognition Through Dynamic Connectionist Models

Sungjae Cho[1] **(sj.cho@snu.ac.kr), Jaeseo Lim**[1] **(jaeseolim@snu.ac.kr),**
Chris Hickey[1] **(chris.hickey@ucdconnect.ie), Jung Ae Park**[2] **(lydia120@snu.ac.kr),**
Byoung-Tak Zhang[1,3] **(btzhang@bi.snu.ac.kr)**
[1] Interdisciplinary Program in Cognitive Science, Seoul National University,
[2] Department of Psychology, Seoul National University,
[3] Department of Computer Science and Engineering, Seoul National University,
1, Gwanak-ro, Gwanak-gu, Seoul, 08826, Republic of Korea

Abstract

The present study aims to investigate similarities between how humans and connectionist models experience difficulty in arithmetic problems. Problem difficulty was operationalized by the number of carries involved in solving a given problem. Problem difficulty was measured in humans by response time, and in models by computational steps. The present study found that both humans and connectionist models experience difficulty similarly when solving binary addition and subtraction. Specifically, both agents found difficulty to be strictly increasing with respect to the number of carries. Furthermore, the models mimicked the increasing standard deviation of response time seen in humans. Another notable similarity is that problem difficulty increases more steeply in subtraction than in addition, for both humans and connectionist models. Further investigation on two model hyperparameters — confidence threshold and hidden dimension — shows higher confidence thresholds cause the model to take more computational steps to arrive at the correct answer. Likewise, larger hidden dimensions cause the model to take more computational steps to correctly answer arithmetic problems; however, this effect by hidden dimensions is negligible.

Keywords: arithmetic cognition; problem difficulty; response time; connectionist model; recurrent neural network; Jordan network; answer step

Introduction

Do connectionist models experience difficulty on arithmetic problems like humans? Although connectionist models consist of abstract biological neurons, similar behaviors between humans and these models are not guaranteed. However, developing model simulations to discover such similarities can bridge this knowledge gap between humans and models, and deepen our understanding of the micro-structures involved in cognition (Rumelhart & McClelland, 1986; McClelland, 1988). Therefore, finding such similarities is a foundational step in understanding human cognition through connectionist models. This connectionist approach recently has been used in the domain of mathematical cognition (McClelland, Mickey, Hansen, Yuan, & Lu, 2016; Mickey & McClelland, 2014; Saxton, Grefenstette, Hill, & Kohli, 2019).

Cognitive arithmetic (Ashcraft, 1992), the study of the mental representation of arithmetic, conceptualizes *problem difficulty*. Problem difficulty can be measured by *response time* (RT) from the time a participant sees an arithmetic problem to the time the participant answers the problem (Imbo, Vandierendonck, & Vergauwe, 2007).

There are three criteria that affect problem difficulty (Ashcraft, 1992; Imbo et al., 2007): (a) operand magnitude (e.g., 1 + 1 vs. 8 + 8); (b) number of digits in the operands (e.g., 3 + 7 vs. 34 + 78); and (c) the number of carry[1] operations (e.g., 15 + 31 vs. 19 + 37). The present study uses a similar experimental approach to that suggested by Cho, Lim, Hickey, and Zhang (2019). This design employs the binary numeral system to control for familiarity with the decimal system and the two criteria (a) and (b). As such, the present study considers the number of carries as the only independent variable involved in problem difficulty.

Recurrent neural networks (Elman, 1990; Jordan, 1997) can model sequential decisions through time. These networks perform sequential nonlinear computations. Owing to the principle that many nonlinear computational steps are required to learn complex mappings (LeCun, Bengio, & Hinton, 2015), parallels can be drawn between human RT and model computational steps in response to problems of varying difficulty level.

Two experiments were conducted in the present study: one on human participants and the other on connectionist models. Both experiments had *learning* and *solving* phases. In the learning phase of the human experiment, participants were taught a method for solving binary arithmetic problems by following guiding examples. In the solving phase, participants began the experiment in earnest, solving arithmetic problems under experimental conditions and having their RTs recorded as a measure of problem difficulty. In the learning phase of the model experiment, connectionist models were trained until they achieved 100% accuracy across all problems. We consider this to be roughly equivalent to how participants were taught to solve arithmetic problems in the learning phase of the human experiment. In the solving phase, all problems were solved again and the number of computational steps taken to solve each problem were recorded as a measure of problem difficulty. Following both experiments, results were analyzed in order to investigate whether any similarities could be observed in how both agents underwent problem difficulty with respect to the number of carries. We then investigated how major model configurations affect model behavior.

[1]A *carry* in binary addition is the leading digit 1 shifted from one column to a more significant column when the sum of the less significant column exceeds a single digit. A *borrow* in binary subtraction is the digit 1 shifted to a less significant column in order to obtain a positive difference in that column. We refer to borrows as carries.

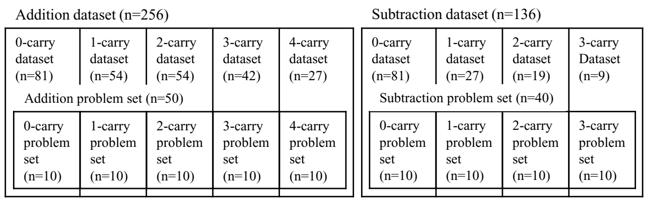

Addition dataset (n=256)
0-carry dataset (n=81)
1-carry dataset (n=54)
2-carry dataset (n=54)
3-carry dataset (n=42)
4-carry dataset (n=27)
Addition problem set (n=50)
0-carry problem set (n=10)
1-carry problem set (n=10)
2-carry problem set (n=10)
3-carry problem set (n=10)
4-carry problem set (n=10)
Subtraction dataset (n=136)
0-carry dataset (n=81)
1-carry dataset (n=27)
2-carry dataset (n=19)
3-carry Dataset (n=9)
Subtraction problem set (n=40)
0-carry problem set (n=10)
1-carry problem set (n=10)
2-carry problem set (n=10)
3-carry problem set (n=10)

Figure 1: Problem sets. The addition and subtraction datasets were assigned to connectionist models. The addition and subtraction problem sets were assigned to participants. *n* refers to the number of operations in a given dataset/problem set.

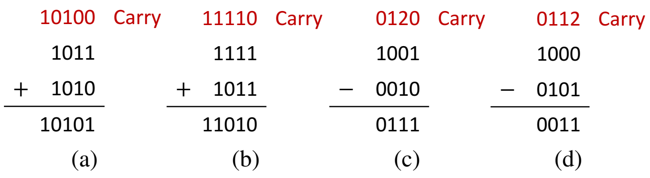


Figure 2: Guiding examples

Problem Sets

Operation datasets For addition and subtraction, we constructed separate *operation datasets*, containing all possible operations between two 4-digit binary nonnegative integers that generate nonnegative results. The addition dataset has 256 operations, and the subtraction dataset has 136 operations (Figure 1). Operation datasets consist of $(\mathbf{x},\mathbf{y})$ where $\mathbf{x}$ is an 8-dimensional input vector that is a concatenation of two binary operands, and $\mathbf{y}$ is an output vector that is the result of computing these operands. $\mathbf{y}$ is 5-dimensional for addition and 4-dimensional for subtraction.

Carry datasets Operation datasets were further subdivided into carry datasets. A *carry dataset* refers to the total set of operations in which a specific number of carries is required for a given operator. The addition dataset was divided into 5 carry datasets, and the subtraction dataset was divided into 4 carry datasets (Figure 1). For example, in Figure 2, the addition guiding examples (a) and (b) are in 2-carry[2] and 4-carry datasets, respectively; the subtraction guiding examples (c) and (d) are in 2-carry and 3-carry datasets, respectively.

Experiment 1: Humans

Experiment 1 investigated whether human RT in problem solving increases as a function of the number of carries involved in a problem.

Participants

90 undergraduate and graduate students (48 men, 42 women) from various departments completed the experiment. The average age of participants was 23.6 ($SD = 3.3$).

Materials

Participants were given two types of problem sets: addition and subtraction. The addition problem set was constructed as follows: 10 different problems were sampled from each carry dataset without replacement[3]. These sampled problems were shuffled together to make the addition problem set. This addition problem set was comprised of 50 unique problems evenly distributed across 5 carry datasets (Figure 1). Likewise, the subtraction problem sets consisted of 40 problems evenly distributed across 4 carry datasets (Figure 1). The problems were newly sampled for each participant.

In any given problem, two operands were presented in a fixed 4-digit format in order to control for possible extraneous influences on problem difficulty as outlined by criterion (b). The experiment was designed in such a way that participants were required to fill out all digits when answering questions (e.g. if the answer was 1, participants were forced to respond with 0001 as opposed to just 1). This is to ensure RT is not affected by the number of answer digits.

Procedure

Participants were shown calculation guidelines containing two guiding examples for addition (Figure 2a, 2b). Participants were explicitly requested to solve problems by using carry operations outlined in the examples. Participants then began to solve each problem from their addition problem set. After solving all addition problems, participants repeated the previous procedure for their subtraction problem set with two subtraction guiding examples (Figure 2c, 2d). Participants were prohibited from using any writing apparatus in order to force participants to solve problems mentally.

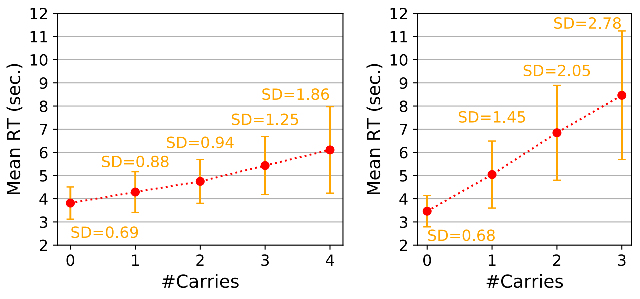
Results

Analysis of variance (ANOVA) was used to investigate differences in mean RTs of participants across carry problem sets. If there were significant differences between all the mean RTs, post hoc analysis was applied. If a participant provided a wrong answer, it was reasonable to assume that this participant made some cognitive error when solving the problem. As such, only RTs for correct answers were included in analysis. We removed the outlying RTs of each carry problem set for each participant since unusually short RTs may be due to memory retrieval and excessively long RTs may be caused by distraction or anxiety during problem solving. The RTs in the range $[Q_1 - 1.5 \cdot \mathrm{IQR}, Q_3 + 1.5 \cdot \mathrm{IQR}]$ were considered outliers, where Q_1 and Q_3 were the first and third quantiles of the RTs for a carry problem set, and $\mathrm{IQR} = Q_3 - Q_1$.

Addition There were significant differences in mean RTs between all carry problem sets, as determined by ANOVA [$F(4,445) = 51.84, p < .001, \eta^2 = .32$]. Post hoc comparisons using the Games-Howell test indicated that mean RTs between any two carry problem sets showed a significant dif-

[2]Let us simply refer to the carry dataset involving *n* carries as the *n*-carry dataset, and problems from the *n*-carry dataset as *n*-carry problems.

[3]This only occurred when sampling 3-carry problems ($n = 10$) from the 3-carry subtraction dataset ($n = 9$). This required one random problem to be duplicated and shown twice in the 3-carry problem set.



(a) Addition (b) Subtraction

Figure 3: Mean RT by carries. The error bars are $\pm 1SD$.

ference [3-carry and 4-carry problem sets: $p = .040$; other pairs: $p < .01$]. Therefore, the mean RT was strictly increasing [4] with respect to the number of carries (Figure 3a).

Subtraction There were significant differences in mean RTs between all carry problem sets, as determined by ANOVA [$F(3,356) = 117.41$, $\eta^2 = .50$]. Post hoc comparisons using the Games-Howell test indicated that mean RTs between any two carry problem sets showed a significant difference [$p < .001$]. Therefore, the mean RT was strictly increasing with respect to the number of carries (Figure 3b).

Experiment 2: Connectionist Models

Experiment 2 investigated whether computational steps required by connectionist models in problem solving increase as a function of the number of carries involved in a problem. Moreover, this experiment intended to examine how the central model hyperparameters — confidence threshold and hidden dimension — affect the simulated RT. The hidden dimension, denoted by d_h, refers to the number of units in the hidden layer.

Model

Imagine the human cognitive process while performing addition and subtraction. Humans predict answer digits one by one while mentally referencing two operands and previously predicted digits. Therefore, we aimed to simulate this human cognitive process by using the Jordan network (Jordan, 1997). The Jordan network is a recurrent neural network whose hidden layer gets its inputs from an input at the current step and from the output at the previous step (Figure 4).

The Jordan network solves problems as follows: An 8-dimensional input vector composed of two concatenated 4-digit operands is fed into the network (Figure 4a). At the same time, its hidden layer with ReLU gets its previous probability outputs. The network predicts step-by-step the probabilities of answer digits up to a maximum of 30 steps (Figure 4b). At the initial step, all digit predictions are initialized as 0.5, which mimics the initial uncertainty humans experience when solving problems. The output layer has sigmoid activation. Each output unit predicts each output digit. The network outputs 5-dimensional and 4-dimensional vectors for addition and subtraction problems respectively.

The network learned arithmetic by minimizing the sum of the losses at all steps: $\sum_t H(\mathbf{z}^{(t)}, \mathbf{p}^{(t)})$. At each step t, a loss is defined as the cross-entropy H between the true answer $\mathbf{z}^{(t)}$ and the output probability vector $\mathbf{p}^{(t)}$ where $\mathbf{x}^{(t)}$ is an input vector: $H(\mathbf{z}^{(t)}, \mathbf{p}^{(t)}) = -\mathbf{z}^{(t)} \cdot \log \mathbf{p}^{(t)} - (1 - \mathbf{z}^{(t)}) \cdot [1 - \log \mathbf{p}^{(t)}]$.

At each time step, the network predicts the probability of every answer digit. When problem solving, humans only decide on an answer digit when they are sufficiently confident that it is correct. Likewise, the network decides each digit only when its predicted probability p_i is higher than some threshold. We call this threshold the *confidence threshold*, denoted by θ_c. Suppose $\theta_c = 0.9$. If a predicted probability p_i is in the range $[0.1, 0.9]$, the model is *uncertain* about the digit. Otherwise, it is *confident* about the digit: if $p_i \in [0, 0.1)$, it predicts the digit is 0; if $p_i \in (0.9, 1]$, it predicts the digit is 1. The network is designed to give an *answer* when it is first confident about all answer digits (Figure 4b). The network in Figure 4b answers at step 1 because this is the first state where the model is confident about all digits. At this answer step, the answer is marked as either correct or incorrect. No answer is given if 30 steps are exceeded.

Measures

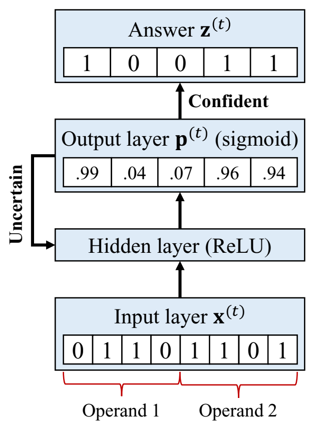
Accuracy *Accuracy* was measured by dividing the number of correct answers by the total number of problems. Model accuracy was used to measure how successfully the model learned arithmetic and to determine when to stop training. No answer after 30 time steps was considered a wrong answer.

Answer step *Answer step* was defined as the index of a certain time step where the network outputs an answer. Answer step is roughly equivalent to human RT. It refers to the number of computational steps required for the network to solve an arithmetic problem. Answer step ranges from 0 to 29.

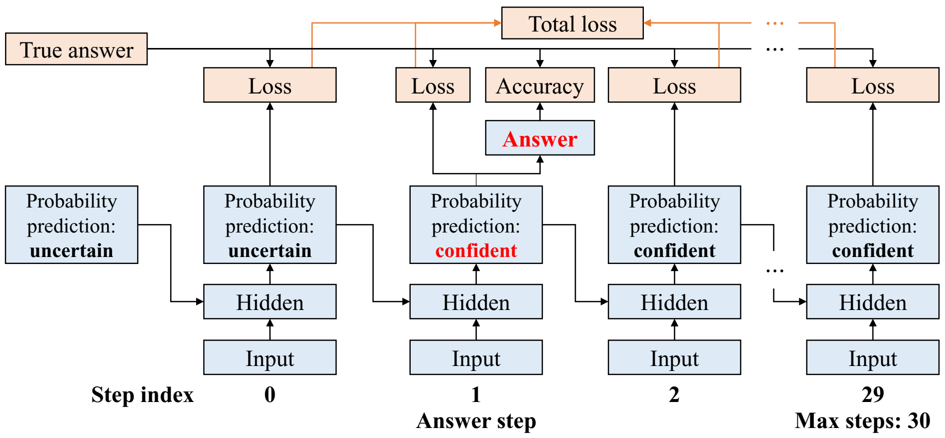
Training Settings

The network learned arithmetic operations using backpropagation through time (Werbos, 1990) and a stocbohastic gradient method (Bottou, 1998) called Adam optimization (Kingma & Ba, 2015) with settings ($\alpha = .001$, $\beta_1 = .9$, $\beta_2 = .999$, $\varepsilon = 10^{-8}$). For each epoch, 32-sized mini-batches were randomly sampled without replacement (Shamir, 2016) from the total operation dataset. The weight matrix $W^{[l]}$ in layer l was initialized to samples from the truncated normal distribution ranging $[-1/\sqrt{n^{[l-1]}}, 1/\sqrt{n^{[l-1]}}]$ where $n^{[l]}$ was the number of units in the l-th layer; All bias vectors $b^{[l]}$ were initialized to 0. After training each epoch, accuracy was evaluated on the operation dataset (Figure 1). When the network attained 100% accuracy for the entirety of the operation dataset, training was stopped. 300 Jordan networks were trained for each model configuration in order to draw statistically meaningful results. Furthermore, to inves-

[4]For every x and x' such that $x < x'$, if $f(x) < f(x')$, then we say f is *strictly increasing*.



(a) The Jordan network for addition



True answer
Total loss
Loss
Loss
Accuracy
Loss
Loss
Answer
Probability prediction: uncertain
Probability prediction: uncertain
Probability prediction: confident
Probability prediction: confident
Probability prediction: confident
Hidden
Input
Step index
0
1
Answer step
2
29
Max steps: 30

(b) The Jordan network unrolled through time steps.

Figure 4: The Jordan network used in the present study. (a) The network is predicting the answer of $110 + 1101$ to be 10011. In this example, the confidence threshold is 0.9. At the current state t, $\mathbf{x}^{(t)} = (0,1,1,0,1,1,0,1)$, $\mathbf{p}^{(t)} = (.99,.04,.07,.96,.94)$, and $\mathbf{z}^{(t)} = (1,0,0,1,1)$. (b) The network is constrained to compute at most 30 steps. The initial probabilities of answer digits are 0.5, meaning the network is uncertain about all digits. The network repeatedly computes the probabilities of answer digits until it becomes confident about all answer digits; in this figure, it answers at step 1. In the learning phase, the network learns from the total loss from all steps. Accuracy is computed by comparing predicted answers to true answers.

tigate if any statistically significant relationship held for various model configurations, we reanalyzed the models with the confidence thresholds $\theta_c \in \{.7, .8, .9\}$ and hidden dimensions $d_h \in \{24, 48, 72\}$. 9 types of networks were trained for both addition and subtraction, respectively; a total of 5400 networks were trained in this experiment.

Results

Our proposed model successfully learned all possible addition and subtraction operations between 4-digit binary numbers. The model required 4000 epochs on average (58 minutes[5]) to learn addition, and 1080 epochs on average (13 minutes) to learn subtraction. When training was completed, we examined: (1) statistical differences in mean answer steps between carry datasets across all model configurations; (2) statistical differences in mean answer steps for operation datasets between different confidence thresholds and hidden dimensions.

Addition The first analysis was conducted on mean answer steps per carry dataset. For every model configuration, ANOVA found significant differences in mean answer steps between all carry datasets (Table 1). Post hoc Games-Howell testing found that for 8 of the 9 model configurations, mean answer step was strictly increasing with respect to the number of carries (Table 1, Figure 5a); the remaining model configuration ($\theta_c = 0.7$, $d_h = 24$) showed a monotonically[6] increasing relationship between mean answer step and the number of carries (Table 1).

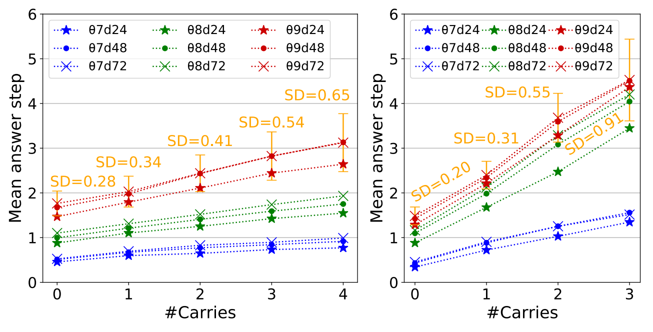
The second analyses were conducted on mean answer steps for the addition dataset. For every hidden dimension, ANOVA found significant differences in mean answer steps between all confidence thresholds $^{\forall}\theta_c \in \{.7, .8, .9\}$ (Table 2). Post hoc Games-Howell testing found that for all models, mean answer step was strictly increasing with respect to confidence threshold (Table 2, Figure 6a). For every confidence threshold, ANOVA found significant differences in mean answer steps between all hidden dimensions $^{\forall}d_h \in \{24, 48, 72\}$ (Table 3). Post hoc Games-Howell testing found that with $\theta_c = 0.7$, mean answer step was monotonically increasing with respect to hidden dimension. For both other confidence thresholds, mean answer step was strictly increasing with respect to hidden dimension (Table 3, Figure 7a). We should note however that while significant, the effect of hidden dimension on mean answer step was small.

Subtraction The first analysis was conducted on mean answer steps per carry dataset. For every model configuration, ANOVA found significant differences in mean answer steps between all carry datasets (Table 1). Post hoc Games-Howell testing found that for all model types, mean answer step was strictly increasing with respect to the number of carries (Table 1, Figure 5b).

The second analyses were conducted on mean answer steps for the subtraction dataset. For every hidden dimension, ANOVA found significant differences in mean answer steps between all confidence thresholds $^{\forall}\theta_c \in \{.7, .8, .9\}$ (Table 2). Post hoc Games-Howell testing found that for all models, mean answer step was strictly increasing with respect to confidence threshold (Table 2, Figure 6b). For every confidence threshold, ANOVA found significant differences in mean answer steps between all hidden dimensions $^{\forall}d_h \in \{24, 48, 72\}$

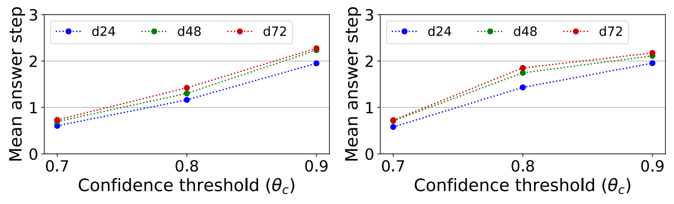
[5]Two Intel(R) Xeon(R) CPU E5-2695 v4 and five TITAN Xp were used. Training networks in parallel is vital in this experiment.

[6]For every x and x' such that $x < x'$, if $f(x) \leq f(x')$, then we say f is *monotonically increasing*.



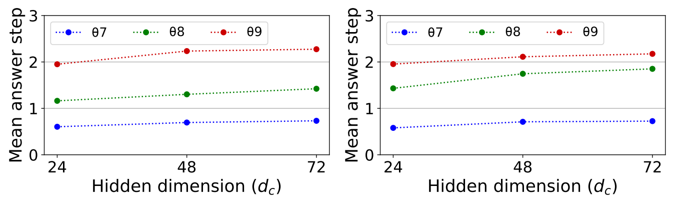
(a) Addition (b) Subtraction

Figure 5: Mean answer step by carries (for carry datasets). θ9d72 denotes models with $\theta_c = 0.9$ and $d_h = 72$. The error bars are $\pm 1 SD$ and belong to θ9d72.



(a) Addition (b) Subtraction

Figure 6: Mean answer step by confidence threshold (for operation datasets)



(a) Addition (b) Subtraction

Figure 7: Mean answer step by hidden dimension (for operation datasets)

(Table 3). Post hoc Games-Howell testing found that with $\theta_c = 0.9$, mean answer step was monotonically increasing with respect to hidden dimension. For both other confidence thresholds, mean answer step was strictly increasing with respect to hidden dimension (Table 3, Figure 7a). We should note however that while significant, the effect of hidden dimension on mean answer step was small (Figure 7a).

Discussion and Conclusion

Experiment 1 Experiment 1 has improved the previous study (Cho et al., 2019) as follows: Firstly, participants were forced to solve problems using solely mental arithmetic. This allows for more valid comparisons to be drawn between humans and models. Secondly, larger data samples allowed the present study to find more statistically significant results. Specifically, mean RT for addition problems were found to be strictly increasing with respect to the number of carries.

Table 1: The results of ANOVA and post hoc analysis on differences in mean answer steps between all carry datasets. The model configuration varies along two axes: confidence threshold and hidden dimension. 300 mean answer steps per carry dataset from 300 trained networks were analyzed for each model configuration. F is the F-test statistic and η^2 is the effect size from ANOVA; in addition, there were 4 degrees of freedom between carry datasets and 1495 within carry datasets: $\mathrm{df}_b^+ = 4$, $\mathrm{df}_w^+ = 1495$; in subtraction, $\mathrm{df}_b^- = 3$, $\mathrm{df}_w^- = 1196$. The mean answer step columns describe the results of post hoc analysis. The inequality ($<$) denotes a significant difference at the $p < .05$ level. Equality ($=$) denotes the opposite. The numbers in these columns refer to the number of carries of a carry dataset. * $p < .05$. ** $p < .01$. *** $p < .001$.

		Addition			Subtraction		
θ_c	d_h	F	η^2	Mean answer step	F	η^2	Mean answer step
.7	24	72^{***}	.16	$0 < 1 = 2 < 3 = 4^{***}$	499^{***}	.56	$0 < 1 < 2 < 3^{***}$
.7	48	206^{***}	.36	$0 < 1 < 2 < 3 < 4^{***}$	765^{***}	.66	$0 < 1 < 2 < 3^{***}$
.7	72	294^{***}	.44	$0 < 1 < 2 < 3 < 4^{***}$	716^{***}	.64	$0 < 1 < 2 < 3^{***}$
.8	24	129^{***}	.26	$0 < 1 < 2 < 3 < 4^{**}$	390^{***}	.49	$0 < 1 < 2 < 3^{***}$
.8	48	198^{***}	.35	$0 < 1 < 2 < 3 < 4^{***}$	571^{***}	.59	$0 < 1 < 2 < 3^{***}$
.8	72	142^{***}	.28	$0 < 1 < 2 < 3 < 4^{**}$	674^{***}	.63	$0 < 1 < 2 < 3^{***}$
.9	24	208^{***}	.36	$0 < 1 < 2 < 3 < 4^{**}$	970^{***}	.71	$0 < 1 < 2 < 3^{***}$
.9	48	421^{***}	.53	$0 < 1 < 2 < 3 < 4^{***}$	1769^{***}	.82	$0 < 1 < 2 < 3^{***}$
.9	72	432^{***}	.54	$0 < 1 < 2 < 3 < 4^{***}$	1718^{***}	.81	$0 < 1 < 2 < 3^{***}$

Table 2: The results of ANOVA and post hoc analysis on differences in mean answer steps between confidence thresholds. $\mathrm{df}_b^+ = \mathrm{df}_b^- = 2$. $\mathrm{df}_w^+ = \mathrm{df}_w^- = 897$. In the mean answer step columns, the numbers refer to confidence thresholds.

	Addition			Subtraction		
d_h	F	η^2	Mean answer step	F	η^2	Mean answer step
24	1032^{***}	.70	$.7 < .8 < .9^{***}$	1163^{***}	.72	$.7 < .8 < .9^{***}$
48	2002^{***}	.82	$.7 < .8 < .9^{***}$	1736^{***}	.79	$.7 < .8 < .9^{***}$
72	1735^{***}	.79	$.7 < .8 < .9^{***}$	1963^{***}	.81	$.7 < .8 < .9^{***}$

Table 3: The results of ANOVA and post hoc analysis on differences in mean answer steps between hidden dimensions. $\mathrm{df}_b^+ = \mathrm{df}_b^- = 2$. $\mathrm{df}_w^+ = \mathrm{df}_w^- = 897$. In the mean answer step columns, the numbers refer to hidden dimension.

	Addition			Subtraction		
θ_c	F	η^2	Mean answer step	F	η^2	Mean answer step
.7	58^{***}	.08	$24 < 48 = 72^{***}$	46^{***}	.10	$24 < 48 < 72^{**}$
.8	38^{***}	.08	$24 < 48 < 72^{***}$	77^{***}	.15	$24 < 48 < 72^{**}$
.9	37^{***}	.12	$24 < 48 < 72^{*}$	51^{***}	.09	$24 < 48 = 72^{***}$

Experiment 2 In Experiment 2, the two hyperparameters — confidence threshold and hidden dimension — were chosen since we expected these hyperparameters to correspond to humans' uncertainty and memory capacity, respectively. We

further expected that increasing confidence threshold and decreasing hidden dimension would increase answer step. This expectation subsequently arose for confidence threshold; confidence threshold had an augmenting effect on answer step. However, our expectation was not born out for hidden dimension. In order to observe clear differences in mean answer steps with respect to problem difficulty, high confidence thresholds are recommended. Hidden dimension should be fixed to the extent that the model can learn an entire dataset.

Experiments 1 & 2 The preceding results show three notable similarities between humans and our connectionist models: Firstly, both agents experienced increased levels of difficulty as more carries were involved in arithmetic problems. Secondly, the Jordan networks with the model configuration ($\theta_c = 0.9$, $d_h = 72$) successfully mimicked the increasing standard deviation of human RT with respect to the number of carries (Figure 3, 5). This phenomenon could not be achieved by a rule-based system performing the standard algorithm, although such a system would be able to simulate increasing RT as a function of the number of carries. Lastly, another similarity found between both humans and models is that the difficulty slope for subtraction is steeper than for addition (Figure 3, 5). This implies that the augmenting effect of carries on problem difficulty is stronger in subtraction than in addition.

Contributions The present study makes two major contributions to the literature: Firstly, our models successfully simulated humans' RT in terms of these three similarities: increasing latency, increasing standard deviation of latency, and relative steepness of increasing latency. The similarities may suggest that some cognitive process, equivalent to the nonlinear computational process used in the Jordan network, could be involved in human cognitive arithmetic. Secondly, the present study demonstrated that fitting our model to arithmetic data induced human-like latency to emerge in the connectionist models (McClelland et al., 2010). In other words, human RTs to arithmetic problems were successfully learned in an unsupervised way. This contrasts with previous studies that focus on learning arithmetic tasks in a supervised way.

Future Study The present study focuses solely on analyzing mean answer steps between arithmetic problem sets of varying difficulty levels. Therefore, future studies could aim to better understand what dynamic processes our model uses when solving individual problems: Specifically, it might be interesting to observe how our model predicts individual digits through each time step when solving problems. Furthermore, similarities between both the model's sequentially predictive answering process and the human answering process could be investigated. This comparison would give us a better understanding of both our model and human mathematical cognition (McClelland et al., 2016).

Our model is designed not just for arithmetic cognition, but also for sequential predictions that based on a constant input and a previous prediction, which result in a single answer. In this regard, this model has the potential to be applied to other cognitive processes involving sequential processing and RT as a measure of cognitive difficulty. Therefore, future studies could consider extending our model to other domains of cognition. For example, well known character image and word classification datasets can be subdivided into datasets of varying difficulty levels, similar to our carry datasets. Mean answer steps for classifying these data sets could be analyzed using a similar model to that outlined in the present study.

Acknowledgments

We thank Seho Park, Seung Hee Yang, Chung-Yeon Lee, Gi-Cheon Kang, and Paula Higgins for useful discussion and writing comments. This work was supported by a grant to Biomimetic Robot Research Center funded by Defense Acquisition Program Administration and Agency for Defense Development (UD160027ID).

References

Ashcraft, M. H. (1992). Cognitive arithmetic: A review of data and theory. *Cognition*, *44*(1-2), 75–106.

Bottou, L. (1998). *Online algorithms and stochastic approximations.* Cambridge University Press.

Cho, S., Lim, J., Hickey, C., & Zhang, B.-T. (2019). Problem diffculty in arithmetic cognition: Humans and connectionist models. In *Proceedings of the 41st annual conference of the cognitive science society.*

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, *14*(2), 179–211.

Imbo, I., Vandierendonck, A., & Vergauwe, E. (2007). The role of working memory in carrying and borrowing. *Psychological Research*, *71*(4), 467–483.

Jordan, M. I. (1997). Serial order: A parallel distributed processing approach. In *Advances in psychology* (Vol. 121, pp. 471–495).

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *3rd international conference on learning representations.*

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436.

McClelland, J. L. (1988). Connectionist models and psychological evidence. *Journal of Memory and Language*, *27*(2), 107–123.

McClelland, J. L., Botvinick, M. M., Noelle, D. C., Plaut, D. C., Rogers, T. T., Seidenberg, M. S., & Smith, L. B. (2010). Letting structure emerge: connectionist and dynamical systems approaches to cognition. *Trends in cognitive sciences*, *14*(8), 348–356.

McClelland, J. L., Mickey, K., Hansen, S., Yuan, A., & Lu, Q. (2016). A parallel-distributed processing approach to mathematical cognition. *Manuscript, Stanford University.*

Mickey, K. W., & McClelland, J. L. (2014). A neural network model of learning mathematical equivalence. In *Proceedings of the annual meeting of the cognitive science society.*

Rumelhart, D. E., & McClelland, J. L. (1986). *Parallel distributed processing* (Vol. 1). MIT Press.

Saxton, D., Grefenstette, E., Hill, F., & Kohli, P. (2019). Analysing mathematical reasoning abilities of neural models. In *7th international conference on learning representations.*

Shamir, O. (2016). Without-replacement sampling for stochastic gradient methods. In *Advances in neural information processing systems 29: NIPS 2016* (pp. 46–54).

Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, *78*(10), 1550–1560.