Problem Difficulty in Arithmetic Cognition: Humans and Connectionist Models

Sungjae Cho[1] (sj.cho@snu.ac.kr), Jaeseo Lim[1] (jaeseolim@snu.ac.kr),
Chris Hickey[1] (chris.hickey@ucdconnect.ie), Byoung-Tak Zhang[12] (btzhang@bi.snu.ac.kr)
[1] Interdisciplinary Program in Cognitive Science, Seoul National University,
[2] Department of Computer Science and Engineering, Seoul National University,
1, Gwanak-ro, Gwanak-gu, Seoul, 08826, Republic of Korea

Abstract

In mathematical cognition, problem difficulty is a central variable. In the present study, problem difficulty was operationalized through five arithmetic operators — addition, subtraction, multiplication, division, and modulo — and through the number of carries required to correctly solve a problem. The present study collected data from human participants solving arithmetic problems, and from multilayer perceptrons (MLPs) that learn arithmetic problems. Binary numeral problems were chosen in order to minimize other criteria that may affect problem difficulty, such as problem familiarity and the problem size effect. In both humans and MLPs, problem difficulty was highest for multiplication, followed by modulo and then subtraction. The human study found that problem difficulty was monotonically increasing with respect to the number of carries, across all five operators. Furthermore, a strict increase was also observed for addition in the MLP study.

Keywords: problem difficulty; arithmetic cognition; binary numeral system; connectionist model; multilayer perceptron

Introduction

Mathematical cognition is the field of research concerned with the cognitive processes that underlie mathematical abilities (Campbell, 2005). Mathematical cognition involves complex mental activities, such as identification of relevant quantities, encoding those quantities into an internal representation, mental comparisons, and cognitive arithmetic. Most notably, cognitive arithmetic is concerned with the mental representation of numbers and arithmetic, and the processes that access and use this knowledge (Ashcraft, 1992).

In cognitive arithmetic, problem difficulty is a central variable (Ashcraft, 1992, 1995). There are at least three criteria for operationalizing problem difficulty: (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 presence or absence of carry[1] operations (e.g., 15 + 31 vs. 19 + 37). In particular, criterion (c) has been further investigated with regard to the number of carries required to correctly solve a problem (Fürst & Hitch, 2000; Imbo, Vandierendonck, & Vergauwe, 2007). In the present study, we investigated how the number of carries affected problem difficulty. *Response time* (RT) from the time a participant sees a problem to the time the participant answers the problem was used in the present study to measure problem difficulty.

[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 $10_{(2)} = 2$ shifted to a less significant column in order to obtain a positive difference in that column. This paper refers to borrows as carries.

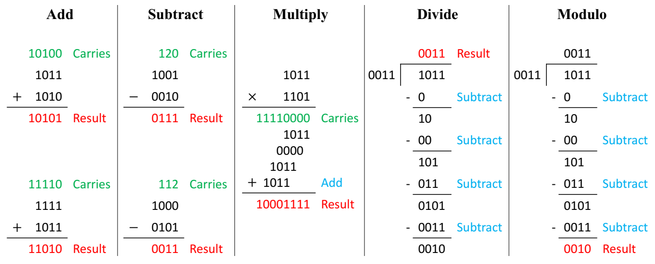


Figure 1: Guiding examples of the five operators with carries.

Previous studies that examine the ways humans process numbers are mostly based on the highly familiar decimal numeral system. Instead, the present study used the binary numeral system, which may offer a novel way to mitigate against the effect of previous experience with conventional mathematical operations. Moreover, since the binary system uses only 0 or 1 digits, it may reduce the *problem size effect*; criterion (a): problems with smaller operands (e.g., $5+2$, $4-1$) are solved more quickly and accurately than problems with larger operands (e.g., $7+6$, $9-6$) (Campbell, 1994; LeFevre et al., 1996; Miller, Perlmutter, & Keating, 1984). Therefore, to observe the effect of carries on problem difficulty, the present study employed the binary system to control for familiarity with the decimal system and criterion (a).

Extending the connectionist approach (Rumelhart & Mcclelland, 1986) to address problems of mathematical cognition may help us understand in detail why mathematics is hard (McClelland, Mickey, Hansen, Yuan, & Lu, 2016). This approach is effective because connectionist models are able to learn many aspects of mathematical cognition. Also, these models offer the possibility to provide concrete instantiations of the mechanisms that grasp the nature of human knowledge and learning within the domain of mathematics.

Previous research has demonstrated how the connectionist model can simulate arithmetic operations. For instance, McCloskey and Lindemann (1992) simulated associative-memory neural networks that learn single-digit multiplication operations. However, these networks were unable to learn all the given arithmetic operations. Utilizing recent advances in deep learning, Kaiser and Sutskever (2016) implemented a recurrent network capable of learning either addition or multiplication of two long binary numbers. This model achieved 100% test accuracy. Franco and Cannas (1998) designed multilayer perceptrons (MLPs) that computed either the addition or multiplication of two binary numbers. The MLPs were constructed with at least one hidden layer and binary step functions as activations. Instead of being learned from data, the weights of the MLPs above were analytically designed. Hoshen and Peleg (2016) made MLPs that learned arithmetic addition, subtraction and multiplication from images of two 7-digit decimal integers through a numerical method.

The MLP was chosen as the connectionist model for the present study due to its strong learning ability, owing to the universal approximation theorem. According to the theorem, an MLP can learn any function if its hidden layers are large enough and its activation functions are squashing functions like sigmoid (Hornik, Stinchcombe, & White, 1989). This implies that MLPs should be capable of learning arithmetic/modulo operations including addition, subtraction, multiplication, division, and modulo[2]. Also, MLPs are a general type of neural networks capable of learning through backpropagation (Rumelhart, Hinton, & Williams, 1986). Based on these properties, we applied the MLP model to help better understand problem difficulty in arithmetic/modulo operations. In order to measure MLP's problem difficulty, we used *conquest epoch*, which is the number of epochs taken by a model to correctly learn a given problem set. We propose this empirical measure since complex nonlinear mappings from harder problems to their correct answers tend to require more epochs than easier problems. In this regard, the conquest epoch can be used to measure the difficulty of learning and solving a particular problem set by MLPs.

Previous studies used one or two arithmetic operators to study problem difficulty. In contrast, the present study investigated problem difficulty across five arithmetic operators[3] — addition, subtraction, multiplication, division, and modulo. The present study also examined problem difficulty across the number of carries for each operator. This provides a more complete view of the impact of both arithmetic operators and carries on problem difficulty. Furthermore, as far as we know, the present study is the first to investigate the impact of operators and carries on problem difficulty in the context of both humans and connectionist models.

Datasets

Operation Datasets For each operator, we constructed an *operation dataset*, containing all possible operations between two 4-digit binary nonnegative integers (ranging $[0, 2^4 - 1]$) that generate nonnegative results. The dataset consists of $(\mathbf{x}, \mathbf{y})$ where $\mathbf{x}$ is an 8-dimensional input vector that is a concatenation of the two operands, and $\mathbf{y}$ is an 8-dimensional output vector that is the result of computing the operands. In Table 1, 'Total' is the number of pairs in each operation dataset. Let us simply refer to, for example, the operation dataset of subtraction as the *subtraction dataset*, and problems from the subtraction dataset as *subtraction problems*. The subtraction dataset is nearly half the size of any other dataset because only problems satisfying $a - b \geq 0$ were included. In the case of division and modulo, the dataset size is $240 = 2^8 - 2^4$ because $a \div b$ where $b = 0$ were excluded. The entirety of these operation datasets was used to train MLPs.

[2]The present study refers to the modulo operation as modulo.

[3]Strictly speaking, modulo is not an arithmetic operator; however, for simplicity, the present study assumes there are five arithmetic operators including modulo.

Table 1: Operation and carry datasets. One operation dataset exists for each operator, and this dataset is subdivided into carry datasets.

	Operation datasets			
# Carries (n)	$+$	$-$	$\times$	$\div$, mod
0	81	81	161	214
1	54	27	11	13
2	52	19	17	9
3	42	9	20	4
4	27		29	
5			5	
6			4	
8			8	
12			1	
Total	256	136	256	240
Carry datasets	5	4	9	4

Carry Datasets Operation datasets were further subdivided into carry datasets. A *carry dataset* refers to the total set of problems requiring a specific number of carries to solve correctly, for a given operator. With n denoting the number of carries required to correctly solve a problem, multiplication has 9 possible n. Hence, multiplication has 9 carry datatsets. The number of carry datasets for the other operators are shown in Table 1. 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*.

Experiment 1: Humans

Participants

153 undergraduate students (89 men, 64 women) from various departments completed the experiment for course credit. The average age of participants was 21.3 ($SD = 1.8$).

Materials

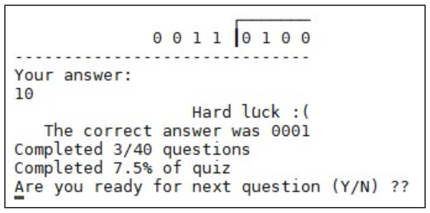
Problem Sets A *problem set* for a specific operator was given to participants. Problems in a problem set were evenly distributed across carry datasets so that participants answered equal numbers of questions from each carry dataset. Question distributions per problem set were as follows: addition – 50 problems across 5 carry datasets; subtraction – 40 problems across 4 carry datasets; multiplication – 45 problems across 9 carry datasets; division – 40 problems across 4 carry datasets;

modulo – 40 problems across 4 carry datasets. Arithmetic problems were randomly sampled from each carry dataset without replacement; let us refer to a set of problems sampled from an n-carry dataset as an *n-carry problem set*. Sampling without replacement prevented participants from answering previously seen problems. However, in rare cases where the number of problems in a specific carry problem set were insufficient[4], participants were presented with the same problem multiple times. Each participant was given a unique randomly sampled problem set. In a given problem, two operands were given in a fixed 4-digit format (Figure 2). This was done in order to control for the extraneous influence of the number of operand digits on problem difficulty, as outlined by criterion (b) in the introduction.

Calculation Guidelines Calculation guidelines were prepared for participants because of their unfamiliarity with the binary system. The guidelines first explained the concept of binary numbers, followed by guiding examples with detailed step-by-step calculations, based on the right-to-left standard algorithm (Wu, 2011). Guiding examples (Figure 1) for each operator were organized as follows: addition – 2 addition problems; subtraction – 2 subtraction problems; multiplication – 1 multiplication problem with 2 addition problems; division – 1 division problems with 2 subtraction problems; modulo – 1 modulo problem with 2 subtraction problems. More than one carry was involved in all guiding examples so that participants grasped the mechanism of carry operations.

Procedure

Participants were randomly assigned to a subset of problems pertaining to one of the five operators; 30 students were assigned to addition, 30 to subtraction, 33 to multiplication, 30 to division, and 30 to modulo. Participants studied detailed calculation guidelines containing one or two guiding examples (Figure 1) for a given operator until they fully understood the given operator. Participants then began the experiment, solving problems through the command line interface (Figure 2). The use of pen and paper was permitted to assist in problem solving. After solving each problem, the true answer was displayed (Figure 2) in order to help participants understand their mistakes and perform more accurately for subsequent problems.



```
              0 0 1 1 |0 1 0 0
------------------------------
Your answer:
10
                    Hard luck :(
   The correct answer was 0001
Completed 3/40 questions
Completed 7.5% of quiz
Are you ready for next question (Y/N) ??
```

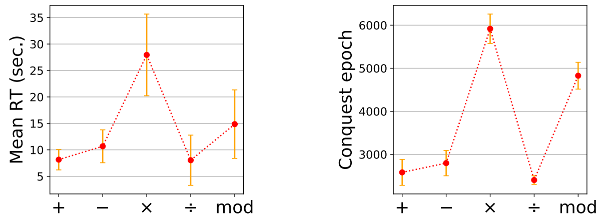
Figure 2: Sample program output.

[4]This was the case for the multiplication 6-carry and 12-carry problem sets, the subtraction 3-carry problem set, and the division/modulo 2-carry and 3-carry problem sets.

Results

If a participant provided a correct answer for a problem, it is reasonable to assume that this participant performed the correct number of carries to arrive at that answer. As such, only RTs for correct answers were used in Experiment 1. Data and detailed analytical results are available in the footnoted repository[5].

Response Time by Operator Each participant's mean RTs across all five operators were analyzed. Let us denote the mean RT for a problem set of operator $*$ as RT^*. Analysis of Variance (ANOVA) was used to investigate differences in RT^* across the five operators $* \in \{+,-,\times,\div,mod\}$. ANOVA showed significant differences between all RT^* [$F(4,148) = 78.65$, $p < .001$, $\eta^2 = .68$]. Further, a post hoc analysis was performed to analyze comparisons between all RT^*. The results of the Games-Howell post hoc test can be denoted by using the following notation[6]: $RT^\times > RT^+$, $RT^\times > RT^-$, $RT^\times > RT^\div$, $RT^\times > RT^{mod}$, $RT^{mod} > RT^+$, $RT^{mod} > RT^\div$ [$p < .001$], $RT^{mod} > RT^-$ [$p < .05$], $RT^+ < RT^-$ [$p < .01$], but $RT^+ \approx RT^\div$, $RT^- \approx RT^\div$ [$p > .05$]. These results can be summarized as: $RT^+ \lessapprox RT^\div \lessapprox RT^- < RT^{mod} < RT^\times$ (Figure 3a).



(a) Humans: Mean RT. (b) MLPs: Conquest epoch.

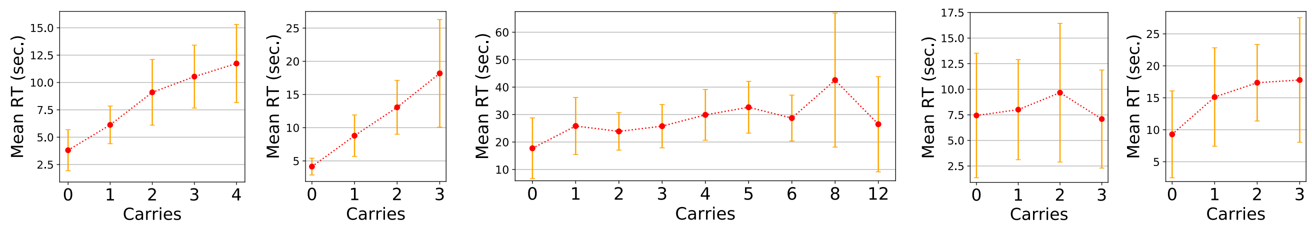
Figure 3: Problem difficulty by operator. The error bars are $\pm 1SD$.

Response Time by Carries Each participant's mean RTs across carry problem sets were analyzed. Let us denote the mean RT for an n-carry problem set of operator $*$ as RT^*_n.

Addition had 5 types of n-carry problems, $n \in \{0,1,2,3,4\}$. ANOVA showed significant differences between all RT^+_n [$F(4,145) = 43.45$, $p < .001$, $\eta^2 = .55$]. The Games-Howell post hoc test revealed that $RT^+_0 < RT^+_1 < RT^+_2$, $RT^+_0 < RT^+_3$, $RT^+_0 < RT^+_4$, $RT^+_1 < RT^+_3$, $RT^+_1 < RT^+_4$ [$p < .001$], $RT^+_2 < RT^+_4$ [$p < .05$], but $RT^+_2 \approx RT^+_3$, $RT^+_3 \approx RT^+_4$ [$p > .05$]. These results can be summarized as: $RT^+_0 < RT^+_1 < RT^+_2 \lessapprox RT^+_3 \lessapprox RT^+_4$ (Figure 4a). As such, for

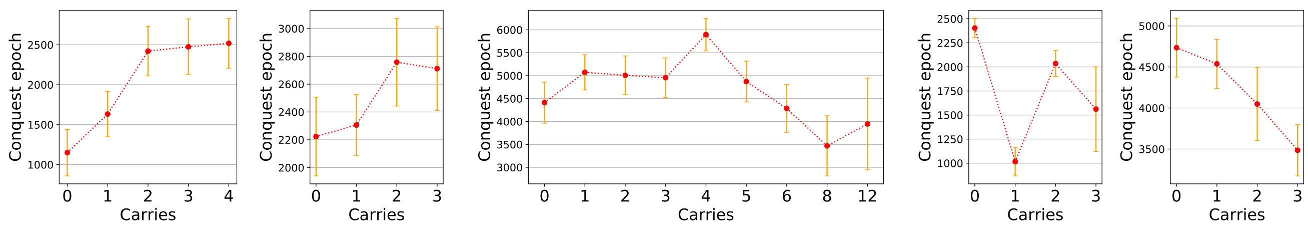
[5]`https://github.com/sungjae-cho/cogsci2019-appendix/tree/master/human`

[6]$A \approx B$ denotes $E[A]$ and $E[B]$ are not significantly different [$p > .05$]. $A < B$ and $B > A$ denote A and B are significantly different [$p < .05$] and their expectations hold $E[A] < E[B]$. $A \lessapprox B \lessapprox C \lessapprox D$ represents $A \approx B$ but A is less than any other right-hand operand (C, D); namely, $A < C$ and $A < D$. Likewise, concerning D, it indicates $C \approx D$, $A < D$ and $B < D$.



(a) Addition. (b) Subtraction. (c) Multiplication. (d) Division. (e) Modulo.

Figure 4: Humans: Mean RT by carries. The error bars are $\pm 1 SD$.



(a) Addition. (b) Subtraction. (c) Multiplication. (d) Division. (e) Modulo.

Figure 5: MLPs: Conquest epoch by carries. The error bars are $\pm 1 SD$.

$n \in [0,2]$, RT_n^+ was strictly increasing[7], but for all n, RT_n^+ was monotonically increasing[8].

Subtraction had 4 types of n-carry problems, $n \in \{0,1,2,3\}$. ANOVA showed significant differences between all RT_n^- [$F(3,116) = 46.07$, $p < .001$, $\eta^2 = .54$]. The Games-Howell post hoc test revealed all pairs of RT_n^- had significant differences. More specifically, $RT_0^- < RT_1^- < RT_2^-$, $RT_0^- < RT_1^- < RT_3^-$ [$p < .001$], $RT_2^- < RT_3^-$ [$p < .05$]. The results can be summarized as follows: $RT_0^- < RT_1^- < RT_2^- < RT_3^-$ (Figure 4b). Therefore, RT_n^- was strictly increasing with respect to the number of carries n.

Multiplication had 9 types of n-carry problems, $n \in \{0,1,2,3,4,5,6,8,12\}$. ANOVA showed significant differences between all $RT_n^\times$ [$F(8,284) = 9.24$, $p < .001$, $\eta^2 = .21$]. The results of the Games-Howell post hoc test can be summarized as follows: $RT_0^\times < RT_3^\times$ [$p < .05$], $RT_0^\times < RT_6^\times$ [$p < .01$], $RT_0^\times < RT_4^\times$, $RT_0^\times < RT_5^\times$, $RT_0^\times < RT_8^\times$ [$p < .001$], $RT_1^\times < RT_8^\times$ [$p < .05$], $RT_2^\times < RT_5^\times$, $RT_2^\times < RT_8^\times$ [$p < .01$], $RT_3^\times < RT_8^\times$ [$p < .05$]; there were no significant difference between the remaining pairs $RT_n^\times$, and only $RT_{12}^\times$ was not significantly different from any other $RT_n^\times$. These results can be summarized as: for $n \in [0,8]$, $RT_n^\times$ was monotonically increasing (Figure 4c).

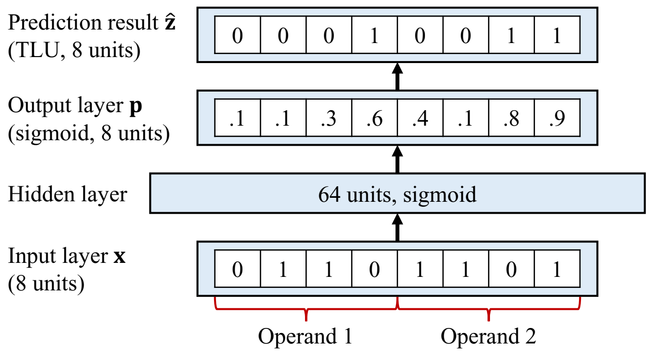
Division had 4 types of n-carry problems, $n \in \{0,1,2,3\}$. ANOVA showed no significant differences between all $RT_n^\div$ [$F(3,116) = 1.20$, $p > .05$, $\eta^2 = .03$]. These results can be summarized as: $RT_0^\div \approx RT_1^\div \approx RT_2^\div \approx RT_3^\div$. Despite no significant difference between any $RT_n^\div$, a weak monotonically increasing trend in mean RT was observable (Figure 4d).

Modulo had 4 types of n-carry problems, $n \in \{0,1,2,3\}$. ANOVA showed significant differences between all RT_n^{mod} [$F(3,116) = 7.78$, $p < .001$, $\eta^2 = .17$]. The Tukey HSD post hoc test revealed that only RT_0^{mod} had significant differences from any other RT_n^{mod}. Specifically, $RT_0^{mod} < RT_1^{mod}$ [$p < .05$], $RT_0^{mod} < RT_2^{mod}$, $RT_0^{mod} < RT_3^{mod}$ [$p < .001$]. These results can be summarized as: $RT_0^{mod} < RT_1^{mod} \approx RT_2^{mod} \approx RT_3^{mod}$ (Figure 4e). RT_n^{mod} was monotonically increasing.

Experiment 2: Connectionist Models

Model

3000 MLPs (Figure 6) were trained for each operator. An 8-dimensional input vector comprised of two concatenated 4-digit operands was fed to the MLP. The MLPs had only one 2^6-unit hidden layer with sigmoid. An 8-dimensional output



Prediction result $\hat{\mathbf{z}}$ (TLU, 8 units)
0 0 0 1 0 0 1 1
Output layer $\mathbf{p}$ (sigmoid, 8 units)
.1 .1 .3 .6 .4 .1 .8 .9
Hidden layer
64 units, sigmoid
Input layer $\mathbf{x}$ (8 units)
0 1 1 0 1 1 0 1
Operand 1
Operand 2

Figure 6: The structure of the multilayer perceptron. The model above predicts that $110 + 1101$ is equal to 10011.

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

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

vector with sigmoid was used in order to match the maximum digit output of the arithmetic results. The predicted result is acquired by processing the output layer through the threshold logic unit (TLU), which transforms output numbers to 1 if they are greater than 0.5, and to 0 otherwise.

Training Settings

MLPs learned arithmetic operations by using backpropagation (Rumelhart et al., 1986) and a stochastic gradient method (Bottou, 1998) called Adam optimization (Kingma & Ba, 2015) with settings: $\alpha = .001$, $\beta_1 = .9$, $\beta_2 = .999$, $\varepsilon = 10^{-8}$. An entire operation dataset was utilized as a training set to train an MLP. For each epoch, a 32-sized mini-batch was randomly sampled without replacement (Shamir, 2016) from the training set. 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 $n^{[l]}$ is the number of units in the l-th layer; all bias vectors $b^{[l]}$ were initialized to 0. The objective function was the sum of the cross-entropy H between the true result $\mathbf{z}(\mathbf{x})$ and output activation vector $\mathbf{p}(\mathbf{x})$ where $\mathbf{x}$ is an input vector from a mini-batch: $\sum_{\mathbf{x}} H(\mathbf{z},\mathbf{p}) = \sum_{\mathbf{x}} [-\mathbf{z}(\mathbf{x}) \cdot \log(\mathbf{p}(\mathbf{x})) - (1-\mathbf{z}(\mathbf{x})) \cdot \{1-\log(\mathbf{p}(\mathbf{x}))\}]$.

For every epoch, accuracy was evaluated on the total operation dataset and each carry dataset (Table 1). When 100% accuracy for a carry or operation dataset was attained, the current number of epochs was recorded as the conquest epoch of the dataset. Training was stopped when 100% accuracy for the operation dataset was reached.

Results

Data and detailed analytical results are available in the footnoted repository[9].

Conquest Epoch by Operator The conquest epochs of MLPs across operation datasets were analyzed. Let us denote the conquest epoch for the operation dataset of operator $* \in \{+,-,\times,\div,mod\}$ as e^*. ANOVA showed significant differences between all e^* [$F(4,14995) = 92838.78$, $p < .001$, $\eta^2 = .96$]. The Games-Howell post hoc test revealed that the differences between all pairs of e^* were significant [$p < .001$]. More specifically, these results can be summarized as: $e^{\div} < e^{+} < e^{-} < e^{mod} < e^{\times}$ (Figure 3b). This mirrors the ordering of the three highest mean RT in humans, as seen in Experiment 1.

Conquest Epoch by Carries The conquest epochs of MLPs across carry datasets were analyzed for each operator. Let us denote the conquest epoch of the n-carry dataset for operator $*$ as e^*_n.

Addition had 5 carry datasets, $n \in \{0,1,2,3,4\}$. ANOVA showed significant differences between all e^+_n [$F(4,14995) = 11835.66$, $p < .001$, $\eta^2 = .76$]. The Games-Howell post hoc test revealed that all pairs of e^+_n were significantly different [$p < .001$]. These results can be summarized as: $e^+_0 < e^+_1 < e^+_2 < e^+_3 < e^+_4$ (Figure 5a). As such, the conquest epoch e^+_n was strictly increasing with respect to n. Again, this mirrors results of RT^+_n from Experiment 1.

Subtraction had 4 carry datasets, $n \in \{0,1,2,3\}$. ANOVA showed significant differences among all e^-_n [$F(3,11996) = 2831.77$, $p < .001$, $\eta^2 = .41$]. The Games-Howell post hoc test revealed that all pairs of e^-_n were significantly different [$p < .001$]. These results can be summarized as: $e^-_0 < e^-_1 < e^-_3 < e^-_2$ (Figure 5b). Therefore, the conquest epoch e^-_n was strictly increasing for both $n \in \{0,1,2\}$ and $n \in \{0,1,3\}$.

Multiplication had 9 carry datasets, $n \in \{0,1,2,3,4,5,6,8,12\}$. ANOVA showed significant differences between all $e^\times_n$ [$F(8,26991) = 5024.17$, $p < .001$, $\eta^2 = .60$]. The Games-Howell post hoc test revealed that differences between all pairs of $e^\times_n$ were significant [$p < .001$]. Specifically, the results can be summarized as: $e^\times_8 < e^\times_{12} < e^\times_6 < e^\times_0 < e^\times_5 < e^\times_3 < e^\times_2 < e^\times_1 < e^\times_4$ (Figure 5c).

Division had 4 carry datasets, $n \in \{0,1,2,3\}$. ANOVA showed significant differences between all $e^\div_n$ [$F(3,11996) = 17788.62$, $p < .001$, $\eta^2 = .82$]. The Games-Howell post hoc test revealed that all pairs of $e^\div_n$ had significant differences [$p < .001$]. More specifically, the results can be summarized as follows: $e^\div_1 < e^\div_3 < e^\div_2 < e^\div_0$ (Figure 5d). Thus, the conquest epoch $e^\div_n$ was not increasing with respect to n.

Modulo had 4 carry datasets, $n \in \{0,1,2,3\}$. ANOVA showed significant differences between all e^{mod}_n [$F(3,11996) = 7281.45$, $p < .001$, $\eta^2 = .65$]. The Games-Howell post hoc test revealed that all pairs of $e^\div_n$ had significant differences [$p < .001$]. The results can be summarized as follows: $e^{mod}_0 > e^{mod}_1 > e^{mod}_2 > e^{mod}_3$ (Figure 5e). Hence, the conquest epoch e^{mod}_n was strictly decreasing with respect to n.

Discussion and Conclusion

Experiment 1 Results of the present study demonstrate how problem difficulty varies depending on the five arithmetic operators and the number of carries. In Experiment 1, results showed that for the five operators, problem difficulty was monotonically increasing with respect to the number of carries (Figure 4). Notably, for subtraction, RT was strictly increasing (Figure 4b). Another notable result was that RT for multiplication was the highest among the five operators (Figure 3a). In order to successfully perform multiplication, several sub-multiplication steps must first be completed (e.g. $1011 \times 1 = 1011$, $1011 \times 0 = 0$, see Figure 1). A participant may have to complete as many as 4-operand addition steps in order to correctly solve a single multiplication problem (Figure 1). It has been shown that the number of steps (DeStefano & LeFevre, 2004) and operands (Seitz & Schumann-Hengsteler, 2000, 2002) involved in arithmetic problems increases working memory demands. As such, the additional arithmetic steps involved in multiplication problems may have led to multiplication having the highest RT among the five operators. It is worth highlighting that participants solved the same 12-carry problem five times, due to

[9] `https://github.com/sungjae-cho/cogsci2019-appendix/tree/master/mlp`

there being only one problem in the 12-carry dataset (Table 1). This problem repetition may be responsible for the decreased RTs seen in the 12-carry problem set, relative to the 8-carry problem set (Figure 4c). As such, it is not valid to compare RT of the 12-carry problem set to other carry problem sets. Like multiplication, modulo problems also require many sub-operations to solve correctly. This may explain why modulo had substantially higher RT than addition, subtraction, and division (Figure 3a). Within the modulo problem set, RT for the 0-carry problems was significantly less than RT for problems involving carries (Figure 4e). However, no significant difference was found in RT between any pair of problem sets involving carries. Modulo involves the use of arithmetic sub-operations in order to correctly answer problems (Figure 1). However, unlike in multiplication, the subtraction sub-operations involved in modulo problems showed consistent patterns. The second operand of the sub-operations was always equivalent to either 0 or the denominator (e.g. 11, see Figure 1). These patterns may have lowered RT for higher *n*-carry datasets (Figure 4e). For division, even if a given problem was an *n*-carry problem, it did not necessarily involve *n* carries, as the final subtraction sub-operation may have been unnecessary in solving the problem (Figure 1). This may have meant that the number of carries in a division problem did not always impact on RT (Figure 4d).

Experiment 2 Experiment 2 found that problem difficulty (conquest epoch) for addition, subtraction, and division was substantially less than problem difficulty for multiplication and modulo. Addition, subtraction, and division may have been easier for MLPs to learn than the other two operators, due to the repeated occurrence of digit patterns in these problems. This implies MLPs learned multiplication and modulo problems by memorizing each problem, rather than by finding digit patterns. This experiment also found that addition problem difficulty for MLPs was strictly increasing with respect to the number of carries involved in a problem (Figure 5a). However, no such increase was seen in the other operators (Figure 5b, 5c, 5d, 5e). Generally, MLPs are sensitive to statistical properties of experience, such as the frequency and typicality of patterns they meet while they learn (Rumelhart & McClelland, 1986). In this regard, MLPs appear to require more epochs to conquer datasets that contain lots of infrequent and atypical patterns. However, the frequency and typicality of patterns in our datasets does not offer a satisfactory explanation as to why an increasing relationship between problem difficulty and carries was observed in addition, but not for the other operators.

Experiments 1 & 2 Comparing Experiment 1 with Experiment 2, humans and the MLPs showed partial similarities in their solving of binary arithmetic problems. For both humans and the MLPs, problem difficulty was highest for multiplication, followed by modulo and then subtraction. (Figure 3). Addition problem difficulty for both humans and MLPs showed increasing trends as a function of the number of carries (Figure 4a, 5a). However, the trajectories of these increases followed notably different paths (Figure 4a, 5a).

Contributions The present study makes four notable contributions to the current literature on mathematical cognition and cognitive science: Firstly, the present study compares problem difficulty across the five operators. This contrasts with preceding work, which has generally dealt with three or fewer operators. Furthermore, to the best of our knowledge, the present study is the first to investigate problem difficulty with regards to the modulo operation. Secondly, the present study for humans showed that the number of carries had a discernible effect on problem difficulty across four of the five arithmetic operators: addition, subtraction, multiplication, and modulo. Thirdly, the use of the binary numeral system allowed the present study to somewhat control for other criteria that may have impacted problem difficulty. These criteria include the problem size effect and over-familiarity with the decimal numeral system. This allowed for a targeted investigation into the effect of carries on problem difficulty. Finally, the present study found that MLPs experienced problem difficulty for some operators similarly to humans: For both humans and MLPs, problem difficulty was highest for multiplication, followed by modulo and then subtraction (Figure 3a, 3b). Also, the effect of carries on problem difficulty in addition problems showed increasing trajectories for both agents (Figure 4a, 5a). This supports previous research (McClelland et al., 2016) suggesting that there may be some similar cognitive processes underlying mathematical cognition in both humans and connectionist models.

Future Study Future studies should aim to uncover what underlying mechanisms caused the MLPs to experience relative problem difficulty similarly to humans across the five operators. Also, the internal representations MLPs use to perform arithmetic operations could be investigated. However, MLPs do not have the innate ability to dynamically process information as humans do. MLPs always take a fixed number of computational steps to produce answers, while humans take a variable amount of time to produce answers. One direction for future work could introduce a new dynamic connectionist model to learn arithmetic, namely, a recurrent network such as the Elman network (Elman, 1990) or the Jordan network (Jordan, 1997). Such recurrent networks can produce answers through variable computational steps depending on the problem. These variable steps can be directly compared to humans' RT, providing a more valid comparison to human arithmetic cognition.

Acknowledgments

We thank Prof. Sungryong Koh for comments on the psychological experiment; Kyoung-Woon On, Gi-Cheon Kang, Taehyeong Kim, Chung-Yeon Lee and Joonho Kim for discussion; Paula Higgins, Seung Hee Yang, and Woosuk Choi for writing comments. This work was partly supported by the Institute for Information & Communications Technology Promotion (R0126-16-1072-SW.StarLab, 2017-0-01772-VTT, 2018-0-00622-RMI, 2019-0-01367-BabyMind) and Korea Evaluation Institute of Industrial Technology (10060086-RISF) grant funded by the Korea government (MSIP, DAPA).

References

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

Ashcraft, M. H. (1995). Cognitive psychology and simple arithmetic: A review and summary of new directions. *Mathematical cognition*, *1*(1), 3–34.

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

Campbell, J. I. (1994). Architectures for numerical cognition. *Cognition*, *53*(1), 1–44.

Campbell, J. I. (2005). *Handbook of mathematical cognition*. Psychology Press.

DeStefano, D., & LeFevre, J.-A. (2004). The role of working memory in mental arithmetic. *European Journal of Cognitive Psychology*, *16*(3), 353–386.

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

Franco, L., & Cannas, S. A. (1998). Solving arithmetic problems using feed-forward neural networks. *Neurocomputing*, *18*(1), 61–79.

Fürst, A. J., & Hitch, G. J. (2000). Separate roles for executive and phonological components of working memory in mental arithmetic. *Memory & cognition*, *28*(5), 774–782.

Hornik, K., Stinchcombe, M. B., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366.

Hoshen, Y., & Peleg, S. (2016). Visual learning of arithmetic operation. In *Proceedings of the thirtieth AAAI conference on artificial intelligence* (pp. 3733–3739).

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).

Kaiser, L., & Sutskever, I. (2016). Neural GPUs learn algorithms. In *4th international conference on learning representations, conference track proceedings.*

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

LeFevre, J.-A., Bisanz, J., Daley, K. E., Buffone, L., Greenham, S. L., & Sadesky, G. S. (1996). Multiple routes to solution of single-digit multiplication problems. *Journal of Experimental Psychology: General*, *125*(3), 284.

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

McCloskey, M., & Lindemann, A. M. (1992). MATHNET: Preliminary results from a distributed model of arithmetic fact retrieval. In *Advances in psychology* (Vol. 91, pp. 365–409). Elsevier.

Miller, K., Perlmutter, M., & Keating, D. (1984). Cognitive arithmetic: Comparison of operations. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *10*(1), 46.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*, 533-536.

Rumelhart, D. E., & McClelland, J. L. (1986). On learning the past tense of English verbs. In *Parallel distributed processing* (Vol. 2, pp. 216–271). MIT Press.

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

Seitz, K., & Schumann-Hengsteler, R. (2000). Mental multiplication and working memory. *European Journal of Cognitive Psychology*, *12*(4), 552–570.

Seitz, K., & Schumann-Hengsteler, R. (2002). Phonological loop and central executive processes in mental addition and multiplication. *Psychological Test and Assessment Modeling*, *44*(2), 275.

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

Wu, H. (2011). The standard algorithms. In H. Wu (Ed.), *Understanding numbers in elementary school mathematics* (pp. 57–60). American Mathematical Society.