A Connectionist Model of Simple Mental Arithmetic

Ivilin P. Stoianov (Ivilin.Stoianov@unipd.it)
Department of General Psychology, via Venezia 8, Padova, Italy

Marco Zorzi (Marco.Zorzi@unipd.it)
Department of General Psychology, via Venezia 8, Padova, Italy

Carlo Umilta (Carlo.Umilta@unipd.it)
Department of General Psychology, via Venezia 8, Padova, Italy

Abstract

This article investigates simple mental arithmetic from a computational perspective and proposes an associative connectionist model that integrates semantic and symbolic representations of numbers. To simulate simple addition, we trained neural networks on addition facts, encoded both semantically and symbolically. Addition tasks were then solved by presenting only the symbolic representations of the operands and retrieving the sum. The networks exhibited the benchmark problem-size effect and tie effect, and accounted for a large proportion of the variance of human addition RTs. Studying the networks during retrieval, we found that they exclusively relied on the semantic “computational core”. We conclude that simple mental arithmetic is a semantic process, and that verbal / Arabic numbers mainly serve as an interface.

Introduction

Simple arithmetic is a principal human numerical ability, thought to have a phylogenetic origin (Butterworth, 1999). For example, pigeons can subtract the numerosity of two sets of objects (Brannon, Wusthoff, et al., 2001) and human infants can sum and subtract small numerosities even before knowing number words (Wynn, 1992). Healthy skilled humans can almost faultlessly add and multiply two single-digit numbers, but crucially, the time course of these operations is systematically affected by their “difficulty”: small problems are retrieved faster than larger problems, what Groen and Parkman (1972) called the “problem-size effect” (Fig. 1). Campbell and Xue (2002) noted that “virtually every study that has examined effects of problem size for simple arithmetic has found that both RTs and error rates tend to increase with the numerical size of the problem.” For competent children and adults, correlations from 0.6 to 0.8 are observed between mean RTs for correct responses and the sum of the operands or the square of the sum, with the latter accounting for a larger proportion of variance (Butterworth, Zorzi, et al., 2001). An additional characteristic of the arithmetic operations is the tie effect: problems such as 7+7 are solved more quickly than non-ties with the same sum (e.g., 9+5) (Groen & Parkman, 1972; Ashcraft & Battaglia, 1978). The tie effect is still poorly understood, with explanations ranging from separate storage (Campbell, 1995) to encoding phenomenon (Blankenberger, 2001).

Various theories struggle to explain the mental operations involved in mental arithmetic, but there is consensus that it is a memory retrieval process (Ashcraft, 1992), probably mixed with procedures when the retrieval fails (Campbell & Xue, 2002). In the “Triple code” model of mental number processing (Dehaene & Cohen, 1995; Dehaene, Piazza, et al., in press), simple addition and multiplication are sets of verbal number facts (such as “two plus five is seven”), even if a domain-specific (semantic) neuronal substrate located in the inferior parietal lobules is considered as underlying other numerical abilities, such as comparison and subtraction.

Figure 1: The problem-size effect (data from Butterworth et al, 2001). Simple-addition RTs of humans (corrected for naming time of the sum) plotted against the sum of the arguments. RTs are best predicted by the square of the sum of the arguments. Additionally, arithmetic problems with equal operands (ties) are solved faster than problems with the same sum but different arguments (the tie-effect).
Associative connectionist theories, on the contrary, share the view that simple mental arithmetic is a number-processing capacity that exploits properties specific to semantic representations of numbers. For example in MATHNET (McCloskey & Lindemann, 1992), which was based on the cognitive model of McCloskey et al. (1985) that postulates that the calculation system deals only with internal semantic representations of number, each digit was encoded with a number line code. This model reproduced the problem-size effect, but only at the cost of implausible fact-frequency manipulation (for fact-frequency data, see Ashcraft & Christy, 1995). Stoianov, Zorzi, et al., (2002) also simulated simple arithmetic with an associative-memory neural network, but they specifically looked for semantic representations that could account for the problem-size effect in realistic learning environments. Among various coding schemes, only Numerosity representations (Zorzi & Butterworth, 1999; see Fig. 2) built upon the cardinal principle could account for the effect. A formal analysis showed that this was due to the joint effect of i) the empirical distribution of active bits (i.e., a bias towards smaller numerosities), and ii) the degree of pattern overlap among arithmetic facts. Another model – Brain-State-in-a-Box (Viscuso et al., 1989) – encoded arithmetic facts in a mixed format, with both magnitude (number-line) and verbal codes. The network learned a sort of approximate multiplication, but its reaction times did not show the crucial problem-size effect.

This paper investigates the nature of the computations underlying simple mental arithmetic by means of simulations with a new connectionist model. Based upon our previous investigations (Stoianov et al., 2002), the model includes a “semantic core” based on Numerosity representations. In addition, since humans normally deal with number facts in verbal or Arabic formats, symbolic representations of the arithmetic facts were also integrated in the input/output representations. The model was implemented using Boltzmann Machine neural networks (Welling & Hinton, 2002) – a distributed associative memory with neurobiologically plausible learning.

We trained the networks to learn the simple addition facts and then studied the computational processes during arithmetic fact retrieval, searching for an explanation of the simulated human capacity. The most crucial finding was that for any kind of input arguments, semantic or symbolic, the network always “calculated” by using the cardinal semantic part. As a consequence, retrieval RTs were determined by the time to encode the arguments and retrieve the semantic result, both of which increase with the numerosity of the arguments. The model also accounted for the tie-effect, produced here due to the variable time to encode symbolic operands into their semantic representations.

**The Model**

The proposed model of simple arithmetic is sketched in Figure 3. The arithmetic module is located in the upper part of the figure. Below it we indicate potential input for numbers: symbolic (e.g., verbal numbers) and semantic (e.g., the numerosity of visual sets). The arithmetic module – the focus of this work – is a distributed associative neural network that in its visible layer encodes the two operands and the result of the arithmetic relationships learned. Based on neuropsychological data (e.g., Cipolotti & van Harskamp, 2001), we propose that different types of arithmetic operations are independently stored. In this article we illustrate the model with additions.

The operands and the result are encoded in both symbolic and semantic forms. In accordance with earlier results that only cardinal semantic organization of numbers can induce the crucial problem-size effect (Stoianov et al., 2002), the semantic operands and result are encoded with the Numerosity code (Zorzi & Butterworth, 1999; see Figure 2) – a linear, discrete implementation of cardinal semantic. As for the symbolic codes, humans deal with more than one type of symbolic representation of numbers – verbal numerals (both spoken and written), Arabic digits, Roman numbers, etc. What is important for this study, however, was how any of these interplay with the semantic code. Therefore, we abstracted symbolic number encoding using a simple two-digit diagonal code, which is independent from the numerical meaning with the exception of the two-digit syntactic structure (the right digit stands for the units and the left one stands for the decades). The distributed associative arithmetic memory was implemented with the Boltzmann Machine NNs (Ackley, Hinton & Sejnowski, 1985) – associative networks of stochastic neurons that iteratively generate patterns according to the distribution of the data learned. They consist of a layer of visible neurons encoding facts and a layer of hidden neurons that learn complex statistical dependencies among the data observed at the visible layer. The networks are fully
connected, without structural biases. To generate patterns, after initialisation of all neurons, BMs iterate until convergence by updating all neurons in parallel, or asynchronously. The number of steps to convergence can be readily interpreted as response time, to be matched against human RT data.

Originally, BMs were trained with a contrastive Hebbian learning algorithm: In a positive step corresponding to classical Hebbian learning, patterns were clamped to the visible layer; the hidden units were settled and the weights were augmented with the mean correlations between every coupled neurons. In a second, anti-learning step, the visible neurons were unclamped; all neurons settled again, and the weights were decreased with the mean correlations for this step. In this way, the visible neurons learned to reproduce the data. However, this stochastic learning algorithm was very slow. Hence, we used the approximate deterministic learning algorithm of Welling and Hinton (2002), which dramatically speeds-up the simulations, maintaining biological plausibility. In our simulations we used this learning algorithm in an unsupervised mode, i.e. there was no input/output distinction (for details, see Stoianov et al., 2002).

Children usually study arithmetic facts in verbal or Arabic notations (but also semantically, e.g. by finger counting or by observing set relations in the visual input). At the time when pupils begin to learn arithmetic, they have already developed, or would shortly have developed, strong associations between the symbolic forms and semantic representations of the numbers. Hence, both the semantic and the symbolic codes would have been activated during learning or practising arithmetic. Accordingly, we trained the networks on both symbolic and semantic input.

In connectionist modelling, one factor that typically affects retrieval of individual patterns is their relative frequencies. Hence, networks were trained in two modes: all facts presented with equal frequencies (to see how factors intrinsic to the model affect its performance), and with fact frequencies extracted from textbooks (data from Ashcraft & Christy, 1995).

After learning, if some of the visible neurons were clamped with a part of a learned pattern, the network would iteratively retrieve the entire pattern according to the data learned. In particular, if we clamped the two arguments, the network would retrieve the result of the corresponding arithmetic operation since in the learning data it was the only completion to this input. Arithmetic problems might be solved with both symbolic and semantic input, but importantly, this could also be done by clamping the symbolic arguments only, retrieving the entire pattern. The latter, in fact, is the best approximation to the task faced by humans when presented with arithmetic problems.

![Figure 3. Computational model of mental arithmetic based on the Boltzmann Machine associative networks (the upper part of the figure; the lower part is given just for reference of possible inputs). Arithmetic facts are encoded at the visible layer in symbolic and semantic formats. The semantic encoding is the cardinal Numerosity code. The model learned the arithmetic facts encoded both in symbolic and semantic formats, developing a fact-retrieval procedure that solved arithmetic tasks semantically even when the problems were presented in symbolic format only.](image-url)
Simulations

Procedure
Simple mental addition was simulated with three learning environments: all addition facts \([1\ldots9 + 1\ldots9]\); the same facts, but with frequencies manipulated according to Ashcraft & Christy (1995), and the facts of an enlarged arithmetic table \([1\ldots12 + 1\ldots12]\). Using the latter was justified by the idea that arithmetic facts with arguments larger than nine could also take a part of the arithmetic knowledge. Indeed, if relations among semantic representations build our arithmetic knowledge and if these codes have a single-bin structure, then there is no need to restrict simple arithmetic to facts with maximal arguments of nine, a limit imposed by the decimal symbolic system that is normally used to form verbal and Arabic numerals.

The performance of trained networks was examined by solving addition problems in two conditions. In the first, both the symbolic and the semantic arguments were set and fixed, allowing the networks to retrieve the semantic and the symbolic representations of the result. In the second, more difficult test, only the symbolic arguments were clamped, allowing the networks to change the semantic arguments and the result. In this case, the result could be retrieved either by using the symbolic associations among the arguments and the result, or by using the systematic relations among the semantic arguments and the result. To exploit the latter, the network would first need to retrieve the semantic arguments. In all tests, the symbolic result was reported.

Performance
The networks learned the addition task in all of the three learning environments. 70% of the facts were learned after 1000 epochs, whereby asymptotic performance (over 95%) was achieved after about 50,000 epochs. The networks first learned to map between semantic and symbolic codes, and to solve addition problems in an approximate manner. The response retrieved after a few learning epochs usually differed just by few units from the result (after 1000 epochs, 70% of the wrong responses were within 2 units from the result), which could only be caused by relying on the systematicity of the cardinal representations. More training resulted in correct answers, either due to refined semantic relations, or because of learned symbolic associations. After learning, the few errors were within one unit from the result.

Reaction Times
The network, tested with problems presented both symbolically and semantically, exhibited the problem-size effect in all learning environments. In the case of uniform fact frequencies, the sum of the arguments predicted 8% of the variance of the network RTs in a linear regression analysis \((p<0.01)\), which was similar to the network with fact-frequency manipulation. Networks trained on enlarged arithmetic table of size 12 produced a stronger effect, with the sum accounting for 24.6% of the variance of the network RTs \((p<0.001)\) and the square of the sum explained 15.5% of the variance \((p<0.001)\). We explain the better fit to the problem-size effect with the specific distribution of the individual bits in the cardinal semantic representations (for details, see Stoianov et al., 2002). Stoianov et al. found similar results in simulations using cardinal semantic codes, but not in simulations using symbolic representations only. The latter produced RTs with a virtually flat distribution. These results strongly suggest that the current model retrieved the results by using the relations among the semantic codes.

Even more significant results were obtained when the networks were tested to solve arithmetic tasks with symbolic arguments only, which they did with nearly the same success (92-94%). We discovered that the network had developed a very specific fact-retrieval procedure, consistently engaged in retrieving the arithmetic facts. It comprised three stages that work in parallel: (i) activating the associated semantic arguments; (ii) retrieving the result corresponding to the current semantic arguments, and (iii), activating the symbolic result corresponding to the current semantic result. The retrieval finishes when all steps converge.

This procedure contributed to producing a better match to the problem-size effect (Figure 4), because the time to access the semantic arguments increased with numerosity, adding up to the time to retrieve semantic facts. In the first training environment, where the facts were presented with uniform frequencies, the sum of the arguments accounted for 81.2% of the variance of the network RTs \((p<0.001)\), while the square of the sum explained 72.2% of the variance \((p<0.001)\). For the training environment where the network learned additions for arguments up to 12, the two structural predictors accounted for 84.6% \((p<0.001)\) and 80.3% \((p<0.001)\) of the variance, respectively.

We also used the network RTs as predictors of the human addition RTs (data from Butterworth et al., 2001). For the simulation with frequency manipulation, the model RTs accounted for 55.9% of the variance in the human data \((N=72, p<0.001)\).
The RTs exhibit the problem-size effect. In addition, ties in general are retrieved faster than non-ties.

The Tie-Effect

The tie effect refers to the phenomenon that arithmetic problems with equal operands (ties) are solved faster than problems with the same sum, but with different arguments (Groen & Parkman, 1972). Our model offers a novel explanation of the tie-effect, based on the finding of Stoianov et al (2002) that one main source of the problem-size effect in associative arithmetic models is the time to activate cardinal codes, whereby time increases with numerosity.

Apparently, this also regards the time to activate the semantic representations of the arguments. As shown before, solving arithmetic problems with symbolic input only requires: a prior activation of the semantic representations of the arguments (i), followed by semantic fact retrieval (ii), and by the activation of the corresponding symbolic result (iii). These steps almost additively affect overall RTs. Let us consider all problems with a given sum \((a+b)\) and the corresponding network RTs. The time for the third step is the same for all problems and can be ignored. The time to retrieve the semantic representation of the same sum \(a+b\) for the various \(a\) and \(b\) can also be regarded as a constant for the present purposes. The remaining source of RT variance comes from the variable time to simultaneously encode the two arguments \(a\) and \(b\). We note that this time is proportional to the magnitude of the larger argument \(RT_{net} \sim max(a,b)\), which is minimized in the case of tie\(^1\). Therefore, we predicted that the model would produce the tie effect when solving arithmetic problems with symbolic input only.

\(^1\)\(\min_{a,b} (max(a,b))\), where \(a+b=const\) is solved by \(a = b = (a+b)/2\).

To test this, we performed multiple regressions on the RTs of the networks of the previous simulations with two predictors: the square of the sum of the arguments – a strong predictor of the problem-size effect – and a binary predictor signaling a tie. For all networks, the tie predictor reached significance, showing shorter RTs in the cases of ties (with betas ranging between -0.14 and -0.12, \(p<0.05\)). We also tested whether the tie effect originated from the addition \(per se\) (step ii). For this purpose, we analyzed the results of the simulations with both semantic and symbolic input, which effectively excluded argument encoding. This time, the regressions with the same predictors showed no significant effect of tie.

We conclude that the source of the tie effect in our model is the argument encoding time. This result is compatible with the finding of Blankenberger (2001) in a recent study with human participants. However, other explanations of the tie effect in humans remain viable.

Discussion

This article presented a connectionist model of simple mental arithmetic that encoded arithmetic facts using both semantic and symbolic representations, an architecture based on the neuropsychological distinction between a domain-specific (semantic) format and verbal / Arabic representations of numbers (Dehaene et al., 2002).

We used the Numerosity code as semantic representations, based on the finding of Stoianov et al. (2002) that only this code could account for the problem-size effect in a connectionist model. Elsewhere, Numerosity has also been shown to account for other basic effects in mental number processing (Zorzi & Butterworth, 1999). We are not aware of any computational study that specifically examined the role of language in the number-processing system. In Stoianov et al. (2002), all attempts to simulate simple arithmetic with symbolic encoding of numbers failed to produce the problem-size effect. The BSB model (Vissuso, et al., 1989) included semantic (number-line) and verbal representations of arithmetic facts, but it did not produce a convincing fit to the problem-size effect (see also a review by Edelman et al, 1996) and the specific roles of the two codes were not investigated.

We trained the model to learn the arithmetic facts with various learning environments. After learning, we discovered that in all testing conditions (with symbolic and semantic encoding of the problems, or only with symbolic encoding) the networks solved the tasks by using the computational “semantic core”. To do so in the second condition, the network first retrieved the semantic representations of the arguments.

The best match to the human problem-size effect was obtained when the networks received symbolic input only – the typical testing condition for humans.
Moreover, the models explained over 55% of the variance of the human RTs in simple mental addition. The symbolic-to-semantic encoding and the "semantic" computation interacted in producing the effect.

Additionally, the model exhibited the tie effect, which has not been studied in earlier computational models. We found that encoding was a source of the effect in the model, a result that parallels the conclusions of Blankenberger’s (2001) study on humans. However, we do not claim that encoding is the only explanation for the tie effect, because addition ties 'a + a' might also be solved with multiplications '2 x a'. This possibility might be verified in a model trained on both additions and multiplications.

The idea that semantic processing is central to mental arithmetic is certainly not new (e.g., McCloskey et al. 1991). Yet, our model is the first complete cognitive modelling in general, because it shows that, 'a + a' might also be solved with multiplications '2 x a'. This possibility might be verified in a model trained on both additions and multiplications.

We consider this a very significant finding for cognitive modelling in general, because it shows that, given an architecture combining systematic semantic representations and associated symbolic ones, the networks learn to use the semantic relations and develop problem-solving "procedures" that access the semantic component even when only symbolic input is provided.

References


