

Review

Can Neural Networks Do Arithmetic? A Survey on the Elementary Numerical Skills of State-of-the-Art Deep Learning Models

Alberto Testolin ^{1,2}¹ Department of General Psychology, University of Padova, 35131 Padova, Italy; alberto.testolin@unipd.it² Department of Mathematics, University of Padova, 35121 Padova, Italy

Abstract: Creating learning models that can exhibit sophisticated reasoning abilities is one of the greatest challenges in deep learning research, and mathematics is rapidly becoming one of the target domains for assessing scientific progress in this direction. In the past few years there has been an explosion of neural network architectures, datasets, and benchmarks specifically designed to tackle mathematical problems, reporting impressive achievements in disparate fields such as automated theorem proving, numerical integration, and the discovery of new conjectures or matrix multiplication algorithms. However, despite this notable success it is still unclear whether deep learning models possess an elementary understanding of quantities and numbers. This survey critically examines the recent literature, concluding that even state-of-the-art architectures and large language models often fall short when probed with relatively simple tasks designed to test basic numerical and arithmetic knowledge.

Keywords: artificial intelligence; neuro-symbolic systems; deep learning; large language models; number embeddings; numerical cognition; mathematical reasoning; arithmetic

Citation: Testolin, A. Can Neural Networks Do Arithmetic? A Survey on the Elementary Numerical Skills of State-of-the-Art Deep Learning Models. *Appl. Sci.* **2024**, *14*, 744. <https://doi.org/10.3390/app14020744>

Academic Editor: Andrea Prati

Received: 5 December 2023

Revised: 12 January 2024

Accepted: 12 January 2024

Published: 15 January 2024



Copyright: © 2024 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Despite the fact that many animal species exhibit an approximate understanding of numbers and quantities [1,2], formal mathematics is a peculiarity of *Homo sapiens* that emerged through thousands of years of cultural evolution [3–5]. Mathematical reasoning requires deploying many finer-grained cognitive abilities, including sophisticated pattern recognition skills, language comprehension, symbolic processing, and abstract thinking, making it one of the highest achievements of human intellect. It is therefore not surprising that the scientific community has always regarded mathematical and logical reasoning as crucial steps towards building intelligent machines [6,7].

However, although computers excel at crunching numbers, solving mathematical problems remains a formidable challenge for artificial intelligence (AI) [8]. On the one hand, grounding structured mathematical knowledge into some form of intrinsic meaning is a long-standing problem in symbolic AI [9–11]. On the other hand, neural networks always struggled in learning math, and such limitations have been traditionally considered an essential feature of their very nature, which is rooted in statistical pattern recognition abilities rather than the use of explicit syntactic rules [12,13]. Mathematical reasoning poses well-known challenges for connectionist models: the symbols used in mathematical formulas appear as arbitrary tokens that need to be manipulated according to well-defined rules entailing compositionality and systematicity. This is difficult to achieve for neural networks, which struggle to process discrete domains since they inherently represent and process information over continuous manifolds [14]. Furthermore, mathematical knowledge extracted from a set of examples should

generalize well beyond the observed distribution, enabling extrapolation through the discovery of “first principles”.

Despite these challenges, the recent successes of deep learning have rekindled interest in the idea that neural networks might be able to acquire high-level reasoning abilities and thus exhibit symbolic behavior [15]. Indeed, although deep networks struggle to grasp even basic concepts such as the meaning of an “integer number” [16], in the past few years, several models have demonstrated impressive capabilities in solving complicated mathematical tasks. For example, sequence-to-sequence architectures were shown able to learn to perform function integration and solve ordinary differential equations, sometimes with greater accuracy than popular math software packages [17]. Deep learning models have also been successfully used for automated theorem proving [18–20] or to assist expert mathematicians to formulate conjectures and establish new fundamental results in pure mathematics [21]. Very recently, deep reinforcement learning discovered a more efficient algorithm to perform matrix multiplication [22], while fine-tuning a pre-trained large language model on computer code allowed to solve university-level math questions at a human level [23].

These stunning achievements partially stem from the introduction of well-curated, large-scale datasets containing mathematical problems annotated with the corresponding solutions, but also from the design of novel (often ad hoc) architectures that can more effectively process numerical symbols and mathematical notation. Improvements in many tasks have also been enabled by the creation of large language models, which exhibit surprising “out of the box” numerical skills that can be further refined through fine-tuning and prompting strategies. However, as we will highlight in the present survey, these findings do not imply that such models fully master the semantics of numbers and basic arithmetic. In fact, their performance on relatively simple numerical tasks is often brittle, and their understanding of symbolic math problems may be much more shallow than it seems [24,25]. This suggests that we might need to improve the elementary skills of these models to bootstrap more reliable mathematical abilities. This hypothesis is also supported by the extended literature on child development and education, which has shown that performance in basic numeracy tasks such as counting, quantity comparison, understanding of number ordering, and the base-ten positional numeral system is a strong predictor of later mathematics achievement [26–28].

The survey is structured as follows (this survey focuses on basic numerical and arithmetic skills; for a more general overview of deep learning models for mathematical reasoning the reader could also refer to [29]): We will initially review the main tasks, datasets, and benchmarks that have been proposed to train and test the arithmetic capabilities of deep learning models. These include numerical problems encoded using natural language (“math word problems”) or simple mathematical formalism (e.g., multi-digit addition and subtraction), as well as other numeracy tasks from basic to advanced levels. We will then present the main neural network architectures that have been proposed to solve this kind of problems, which include ad hoc modules specifically tailored to process mathematical notation, but also general-purpose deep learning systems and large language models. We will also review the main strategies that have been adopted to inject number semantics into word embeddings and provide some representative examples of arithmetic calculation performed by state-of-the-art AI models, such as GPT-4.

2. Elementary Numerical Tasks and Datasets

2.1. Math Word Problems

Math word problems (MWP) are rapidly becoming a standard benchmark for AI models (for a recent review, see [30]). They are closed-domain, narrative problems that are used to assess the general understanding of numbers and operations through everyday life situations and are one of the most common types of numerical task

encountered by children in primary schools. MWP are framed in natural language and vary in complexity depending on the type of arithmetic operations they require, and whether such operations involve small or large operands.

Most MWP datasets used in AI research are either curated from educational or online resources, derived by processing other available datasets, or synthetically created. One of the first large-scale datasets was Dolphin18K [31]; it contains 18,000 elementary math word problems taken from an online math repository, annotated with ground truth information using a semi-automatic procedure. A similar dataset is Math23K [32], which contains 23,161 MWPs in the Chinese language crawled from online education websites. A step forward in dataset creation was implemented in the AQuA dataset [33], which contains 100,000 problems annotated not only with the correct answer, but also with the sequence of steps (“answer rationale”) required to derive the solution. Problems were generated by starting from a set of seed problems, which were then modified through crowdsourcing. This approach for building a large-scale MWP dataset was further refined in MathQA [34] by introducing a more precise language to define operation programs representing the intermediate problem solution steps. It contains 37,200 problems taken from the AQuA dataset, annotated with the corresponding lists of multiple-choice options and aligned operation programs, again using a crowdsourcing platform. An example of such problems is given in Table 1A.

In most of the cases discussed above, the authors implemented some control procedures to limit the possibility that crowdsources could create duplicate versions of the same problem (e.g., by copying and pasting online problems or by proposing trivial modifications to the problem text). However, the research community recently pointed out that these datasets actually contain many problems with an overlapping structure or very similar content [35]. This makes a large fraction of MWPs solvable through simple heuristics, for example, by treating the text as a bag of words or even ignoring the question during the generation of the answer, leading to a significant overestimation of model performance [36]. These shortcomings called for the design of more controlled datasets. One of them is ASDiv [35], which contains only 2305 problem instances; however, they are provably more heterogeneous compared to previous corpora thanks to the use of a metric introduced to measure the lexicon usage diversity. This design principle was also adopted for the creation of a larger-scale benchmark called GSM8K [37], which contains 8500 high-quality math problems at the elementary school level, created by human problem writers. This dataset was similarly built with the goal of featuring high linguistic diversity, while relying on relatively simple math concepts: problems take between two and eight steps to solve, and solutions primarily involve performing a sequence of elementary calculations using basic arithmetic operations. Another recently proposed benchmark is SVAMP [36], a dataset containing 1000 simple (one unknown) arithmetic word problems of a grade level of up to 4. It was created by applying a set of variations to existing problems sampled from the ASDiv dataset that were carefully designed to probe whether the model’s answer actually depends on the question (“question sensitivity”), whether the model has the capacity to determine a change in reasoning arising from subtle changes in the problem text (“reasoning ability”), and whether the answer remains invariant to superficial structural changes that do not alter the problem semantics (“structural invariance”).

An alternative recent benchmark relies on “feasibility” questions to investigate numerical reasoning involving approximate magnitude information [38]. In this case the model is probed with binary or multiple-choice questions that require identification of which of the alternatives provided are compatible with the numerical magnitudes described in the problem (an example is reported in Table 1B).

2.2. Operational Problems Encoded Using Simple Mathematical Notation

Other benchmarking approaches directly probe mathematical knowledge using numerical symbols and formal notation, with the goal of testing whether AI models can

systematically learn to solve problems with compositional structure. In this case, the challenge is mostly to demonstrate that the model can extrapolate beyond the range of numbers and operations encountered during training, for example, by counting a greater number of items or solving arithmetic problems with more (and possibly longer) operands.

In this setting, test problems are normally generated using synthetic procedures. One landmark study [16] proposed to evaluate neural networks using simple function learning tasks (i.e., arithmetic operations), in some cases requiring an initial step of perceptual processing, as in the “MNIST Digit Counting” and “MNIST Digit Addition” tasks. Model performance is then assessed on held-out values from within the training range (interpolation) or from outside of the training range (extrapolation). A similar procedure was also adopted in more recent studies, with the goal of systematically characterizing extrapolation capabilities across numerical ranges [39–41] or to investigate “length generalization” as the ability to extrapolate from short problem instances to longer ones [42]. Another class of basic tasks that are particularly challenging in the extrapolation regimen involves the translation from number words to quantities, as in the “language to number translation task” [16] or, vice versa, from quantities to number words. These tasks probe whether the compositional structure of number words and number symbols is learned in a systematic way. An example of such problems is given in Table 1C.

One of the most popular synthetic benchmarks containing math problems encoded using formal notation is the Mathematics dataset [43]. It was built with the main goal of providing a controlled set of problems, for example, by making sure that test problems contained questions that were never seen during training. The generation method allows problems to be classified according to math domain, difficulty, and the need to perform interpolation or extrapolation. The dataset contains two million free-form question/answer pairs, encoded as sequences of characters, spanning problems in algebra, arithmetic, calculus, number comparison, measurement, and probability. Extrapolation capabilities can be measured along different axes, for example, by introducing problems involving larger numbers, more numbers, and more compositions of basic operations. Examples of such problems are given in Table 1D.

2.3. Higher-Level Numeracy and Math Problems

Open-domain questions involving numerical reasoning have also recently been included in general “reading comprehension” benchmarks [44,45]. Compared to MPWs, this kind of benchmark usually contains much longer language contexts and requires deeper paragraph understanding, for example, by entailing numerical reasoning over dates or ordering of events in the text paragraph. An example of such problems is given in Table 1E. One challenging benchmark of this kind is NumGLUE [46], which contains approximately 100,000 problems covering eight different types of tasks, all of which require simple arithmetic understanding. Some problems are self-contained, while others require additional background knowledge to produce the final solution, such as commonsense reasoning (e.g., “How many faces do 10 dice have?”).

Another recently introduced dataset is MATH [47], which consists of 12,500 problems taken from high-school math competitions that span a variety of math domains, carefully annotated with step-by-step solutions. Given the difficulty of these problems, the authors also released a large-scale pretraining dataset called Auxiliary Mathematics Problems and Solutions (AMPS), which contains over 100,000 Khan Academy problems with step-by-step solutions in Latex format; these exercises are used to teach human students concepts ranging from elementary math to multivariate calculus.

The authors of NumGLUE also recently proposed LILA [48], a “unified” mathematical reasoning benchmark consisting of 23 mathematical reasoning tasks. It has been built by extending 20 existing datasets spanning a wide range of topics in mathematics, matched (in a semi-automatic way) with corresponding Python programs that serve as reasoning chains for each question in the benchmark. The authors also

include an additional benchmark dataset to measure performance specifically on out-of-distribution examples and to test model robustness to language perturbations, like in [36].

Table 1. Examples of problems from representative numerical reasoning datasets. (A) MathQA [34]. (B) FeasibilityQA [38]. (C) Language to number translation task [16]. (D) Mathematics [43]. (E) NumGLUE [46].

A Problem:	Bruce purchased 7 kg of grapes at the rate of 70 per kg and 9 kg of mangoes at the rate of 55 per kg. How much did he pay to the shopkeeper?
Operation stack:	a = multiply (7, 70); b = multiply (9, 55); add (a,b)
Answer:	985
B Problem:	Sam gave 50 dollars to the shopkeeper to buy a book and the shopkeeper returned some money. What could be the price of the book? Choose all plausible alternatives.
Answers:	(a) 55 (b) 45 (c) 37 (d) 59 (e) None
C Problem:	Write the Arabic number corresponding to the following number words: "three hundred and thirty-four" "one thousand two hundred and seven" "eighty-nine thousand and one"
Answer:	334; 1207; 89001
D Problem:	Divide 1136975704 by -142121963.
Answer:	-8
Problem:	Let $k(u) = u^2 + u - 4$. Find $k(0)$.
Answer:	-4
E Context:	Before the UNPROFOR fully deployed, the HV clashed with an armed force of the RSK in the village of Nos Kalik, located in a pink zone near Sibenik, and captured the village at 4:45 p.m. on 2 March 1992. The JNA formed a battlegroup to counterattack the next day.
Question:	What date did the JNA form a battlegroup to counterattack after the village of Nos Kalik was captured?
Answer:	3 March 1992

A further step forward has been taken by MathVista [49], which is a newly introduced benchmark designed to combine challenges from diverse mathematical and visual tasks, derived from 28 existing multimodal datasets involving mathematics and 3 newly created datasets. Problems in MathVista require visual understanding and compositional reasoning over a variety of numerical and mathematical domains. Some of them focus on basic numeracy skills (e.g., length estimation, number comparison, approximate calculation, etc.); however most problems cannot be considered "elementary" since they involve sophisticated mathematical structures that probe abstract reasoning.

Overall, the performance level on these more challenging benchmarks is far from ceiling and most neural network models perform significantly worse than humans [50,51]. However, it should be noted that these datasets are also challenging for humans; university students have been estimated to reach around 40% on MATH, while a three-time IMO gold medalist attained 90% [47].

3. Neural Network Models for Arithmetic Reasoning

3.1. Ad Hoc Architectures

Several frameworks have been proposed to create "neuro-symbolic" systems that combine neural networks with rule-based numerical solvers or to design neural network

modules specifically tailored for numerical reasoning. One of the earliest attempts to solve math word problems using a neuro-symbolic approach consisted of a vanilla sequence-to-sequence model combined with a similarity-based retrieval model, with the goal of creating a hybrid MWP solver [32]. Subsequent approaches exploited graph-based problem representations, which facilitate the solution of MWP through the use of structured descriptions reflecting a tree-like algorithmic decomposition of the problem [52,53] or explicitly capturing the relationship between quantities and their attributes [54]. Another ad hoc architecture incorporates numerical reasoning into the system by exploiting a “numerically aware” graph neural network [55]. The model is composed by standard neural encoding and decoding modules, but also includes a “reasoning module” that represents the MWP quantities using a graph, where the nodes correspond to the numbers appearing in the text and the edges encode magnitude relationships (i.e., “greater than”). However, since the graph is pre-defined for each problem, such a model cannot deal with tasks entailing the generation of intermediate numerical results and has limited arithmetic reasoning abilities. Overall, we can argue that models incorporating ad hoc graph-based representations generally improve over standard seq2seq architectures, but their performance is still poor when tested using more controlled (though relatively simple) datasets [35,36].

A modular design was also adopted by a recent neuro-symbolic model developed to tackle reading comprehension problems involving numerical reasoning [44]. In this architecture, a transformer-based module first predicts whether the problem answer is a count or an arithmetic expression, and then identifies the specific numbers involved in the expression. Once a proper arithmetic expression has been formed, it is given as an input to a symbolic solver to produce the final answer. The model improved over the considered baselines; however, its testing accuracy was still far from the human level in the dataset considered by the authors (44% vs. 94%). Another recent approach proposed to extend the regular transformer architecture to promote problem decomposition into reusable building blocks, applied in sequential steps. This allowed a significant increase in performance in terms of simple arithmetic and list processing tasks [56]. A different modular neuro-symbolic architecture was shown to be able to improve on out-of-distribution nested arithmetical expressions when trained only on a subset including the simplest cases, by iteratively applying a learned substitution rule on the input string [57].

Inspired by the fact that neural networks fail to learn to represent numbers outside of the range seen during training, others proposed to augment standard artificial neurons with ad hoc modules biased to learn systematic numerical computation. For example, the “neural arithmetic logic unit” is augmented with operators that can represent simple functions such as addition, multiplication, and power functions [16]. The “neural arithmetic unit” generalizes this idea and achieves higher extrapolation capabilities by introducing a simplification of the parameter matrix, a sparsity regularizer, and a new multiplication unit [41]. In general, these models demonstrate some extrapolation capabilities in simple arithmetic tasks, for example, by accumulating moderate test errors on additions involving two 1000-digit numbers when being trained only with 10-digit numbers. However, accuracy is still far from ceiling, and these models do not easily generalize to problems involving a higher number of operands.

3.2. Generic Deep Learning Architectures

Many researchers have also tried to tackle mathematical reasoning by exploiting generic neural network architectures, hypothesizing that learning to perform numerical reasoning might not be qualitatively different from the acquisition of domain-general “abstract reasoning” abilities and should thus be treated as a general (though particularly challenging) learning problem. However, it has been repeatedly pointed out that generic neural network architectures struggle in the extrapolation regimen [16], with sequence-to-sequence or even advanced transformer-based architectures often failing in tasks requiring intermediate calculations [43]. The related question of whether and how

recurrent networks generalize to sequences longer than those encountered during training has also been of enduring interest. Definitive solutions to such a challenging issue have not yet been found [42], although recent investigations have shown that relative position embeddings facilitate out-of-distribution length generalization in simple addition tasks [58].

The research community has been actively exploring novel domain-general neural architectures that might overcome these limitations. One interesting approach to tackle algorithmic learning (a.k.a. “program synthesis”) has been to augment generic deep learning architectures with external memory modules, which enable systematic abstraction even from few examples and can (at least theoretically) learn to approximate any algorithmic procedure [59,60]. Interestingly, neural models equipped with external memory have indeed been shown as able to generalize well beyond their training range in binary addition and multiplication problems [61]. Subsequent work has further refined these results by improving the format of the external memory, for example, by showing that a grid-like memory representation can significantly improve out-of-distribution generalization on decimal addition, both in terms of digit length and number of operands [40,62].

The idea of granting access to an external memory to solve complex problems is reminiscent of the notion of “material representation” introduced in anthropology, which has been recently elaborated on in the context of numerical cognition [63]. According to this view, abstract mathematical concepts would be a relatively recent cultural achievement, which have emerged thanks to the spread of numerical manipulation tools [64]. This perspective has recently been explored in computational models based on deep reinforcement learning, which can simulate the active interaction of a learning agent with external numerical representation devices [65,66]. Related streams of research seek to improve deep learning models by granting them access to external tools [67], constructing modular architectures equipped with discrete knowledge bases and reasoning components [68], or creating multi-modal systems that can ground numerical symbols into visuo-spatial representations [49].

3.3. *Generic Large Language Models*

One of the most exciting discoveries of the past few years has been to realize that large language models (LLMs) trained in an autoregressive way can exhibit a surprising level of competence in a variety of tasks “out of the box”, that is, without having been explicitly trained to solve such problems. This also seems to be the case for numerical reasoning; for example, GPT-3 is able to carry out basic arithmetic tasks such as two-digit addition without any additional fine-tuning, with performance becoming progressively stronger moving from the zero-shot to one-shot to few-shot setting [69]. However, the numerical knowledge of LLMs is often superficial and inaccurate: the calculation accuracy of GPT-3 rapidly decreases as the number of digits increases [69] and further growing the model size does not necessarily improve the extrapolation capabilities [70,71]; the text-to-text T5 model fails in simple numerical tasks, such as number comparison and number sorting, when probed outside the interpolation range [72]; and the mathematical performance of the popular ChatGPT model is significantly below that of an average mathematics graduate student [51]. In general, even the largest and most recent models falter when required to perform multi-step mathematical reasoning, such as those included in the MATH benchmark. It has also been shown that the performance of LLMs on mathematical calculations strongly correlates with term frequency in the training data [73], suggesting that these gigantic models might obtain a seemingly high performance from surface-level memorization, rather than an understanding of arithmetic procedures.

One possibility to improve the numerical reasoning in LLMs is to fine-tune them using domain-specific datasets. For example, GPT-style models can solve many of the problems in the Mathematics benchmark once trained on math datasets [70], and fine-tuned BERT, T5, or LLaMA models also achieve significant improvements on more

challenging benchmarks [74–77]. Yet, fine-tuned LLMs still fail, even in simple numerical tasks that probe extrapolation capabilities, and sometimes performance is even worse than the original architectures [72].

Performance can also be improved by combining generative LLMs with post-processing verifiers, which are trained to judge the correctness of model-generated solutions: at test time, a fixed number of candidate solutions is sampled and the one that is ranked highest by the verifier is selected as the final output [37]. In this specific case, the system was also further trained to rely on a calculator for solving arithmetic problems, by injecting calculation annotations into the training set; at test time, the calculator overrides sampling when the model produces the annotations. The idea of producing annotations that can be subsequently processed by a software calculator is also pursued in more sophisticated approaches that train LLMs to generate computer code to solve complicated algorithmic tasks, rather than directly asking for the solution. In one recent demonstration of this method [23], the authors exploited an LLM pre-trained on text and fine-tuned on computer code [78] to generate computer programs that solve challenging (university-level) problems from the MATH benchmark. The performance gain was significant, though it should be emphasized that in these cases, the task of producing the final solution is partially delegated to an external tool (e.g., a Python interpreter), which can also take advantage of numerical libraries to perform mathematical operations such as solving equations or taking limits.

3.3.1. Promoting Step-by-Step Numerical Reasoning

A promising approach to improve the numerical skills of LLMs involves the use of advanced prompting strategies (a.k.a. “in-context learning”), which allow the model’s behavior to be shaped by providing a few input–output exemplars demonstrating the task, without actually tuning any model parameter [69]. A recent line of study has shown that prompting strategies can be improved using “scratchpads” [79] and “chain-of-thought reasoning” [80], which allow accuracy to be significantly increased on multi-step mathematical reasoning problems. The idea is that rather than having to generate a final answer immediately, the model can first generate solutions that may contain intermediate computations (see Table 2). To achieve this, the scratchpad technique introduced by [79] allows the model to produce an arbitrary sequence of intermediate tokens that are stored in a buffer and that can be further processed before producing the final answer. The authors considered the task of learning long integer addition, showing that the use of scratchpads improved the calculation accuracy in the extrapolation interval. This idea was taken further by [80]: the “chain-of-thought” is a series of intermediate natural language reasoning steps that are provided as an input to the model during prompting to explicitly demonstrate how to produce the final output. This technique was tested in arithmetic reasoning tasks, achieving striking performance gains compared to baseline models. Interestingly, it turns out that chain-of-thought prompting has larger performance gains for more complicated problems, mostly when it is applied to larger-scale models. At the time of publication, it established a new state-of-the-art performance on the challenging GSM8K and SVAMP benchmarks.

Table 2. Examples of numerical problems solved using advanced prompting strategies. (A) The scratchpad method forces the model to explicitly produce intermediate computation steps, which are iteratively refined to generate the final answer [79]. (B) A subset of chain-of-thought prompting exemplars that were given as input to the model to elicit step-by-step reasoning during problem solution [80].

A Problem:	29 + 57
	29 + 57, C: 0
Scratchpad:	2 + 5, 6 C: 1 # added 9 + 7 = 6 carry 1
	8 6 C: 0 # added 2 + 5 + 1 = 8 carry 0

	0 8 6
Answer:	86
B Problem:	If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?
Chain-of-thought:	There are originally 3 cars. 2 more cars arrive. $3 + 2 = 5$.
Answer:	The answer is 5.
Problem:	There were nine computers in the server room. Five more computers were installed each day, from Monday to Thursday. How many computers are now in the server room?
Chain-of-thought:	There were originally 9 computers. For each of 4 days, 5 more computers were added. So $5 * 4 = 20$ computers were added. $9 + 20$ is 29.
Answer:	The answer is 29.

Similarly to what happens in recurrent models with an adaptive computation time [81,82], these advanced prompting techniques allow the neural network to process the input information for as long as is needed, depending on the complexity of the current problem. By encouraging the model to produce an explanation along with the answer, we also steer it towards solving problems by breaking them into smaller steps that logically follow from each other. Furthermore, the buffer allows intermediate information to be stored for an arbitrary amount of processing steps, removing the need to memorize all intermediate states in the network's activations. Last, but not least, this approach provides an interpretable window into the behavior of the model, suggesting how it might have arrived at a particular answer and providing opportunities to debug where the reasoning path went wrong. In fact, structured prompting techniques are reminiscent of the educational process adopted by teachers in primary schools, where the elementary steps for solving algorithmic problems (such as carrying out long additions) are explicitly written out to facilitate understanding.

These methods have only scratched the surface of the potential of prompting for eliciting high-level reasoning. Minerva [50], which is a version of the PaLM model, fine-tuned on a high-quality dataset containing scientific and mathematical data, exploits chain-of-thought prompting and generates multiple candidate solutions that are selected using a majority voting scheme. This allows this model to significantly outperform the original PaLM model, although its average accuracy remains below the human level. Other recent improvements introduced algorithmic prompting methods [83,84], which provide a detailed description of the algorithm execution on running examples and use explicit natural language instructions to further remove ambiguity. This increases the amount of detail included in the input rationales, whilst at the same time, promoting the composition of basic skills to solve complex reasoning tasks. Such prompting strategies can outperform standard prompting techniques on several algorithmic tasks, including arithmetic [85], and achieve a much higher performance on addition problems, also in the extrapolation range [83]. At the moment, one of the most advanced LLMs is GPT-4 [86], which indeed, set a new bar in all benchmarks related to mathematical reasoning and numerical manipulation. However, it is interesting to note that even GPT-4 sometimes makes elementary mistakes if not correctly prompted (see example in Table 3).

Table 3. Example of interaction with GPT-4. The model gives a wrong response when directly asked for the result of a simple arithmetic problem, while it produces the correct answer when forced to also report the intermediate calculation steps [test performed on 1 December 2023].

USER:	What is the result of the following expression? $3 * 7 + 9 * 8 + 9 =$
ASSISTANT:	106

USER:	What is the result of the following expression? Please report also the intermediate calculation steps. $3 * 7 + 9 * 8 + 9 =$
ASSISTANT:	First, perform the multiplication operations: $3 * 7 = 21$ $9 * 8 = 72$ After that, add the results along with the 9: $21 + 72 + 9 = 102$ So, the result of the expression $3 * 7 + 9 * 8 + 9 =$ is 102.

3.3.2. Injecting Numerical Semantics into Word Embeddings

It turns out that the way numbers are represented in their “surface form” has a strong influence on the model performance in numerical reasoning tasks. Although some initial investigations suggested that standard embedding techniques could capture number semantics fairly well [87], subsequent studies pointed out that these representations are in fact inadequate at dealing precisely with numbers [88] and that commonly used sub-word tokenization techniques disrupt magnitude information [89]. For example, in GPT-3, a number like 1598 is tokenized as “15” and “98”, while another format like 1,598 is split as three different tokens: “1”, “,”, and “598”.

Simple tricks to improve model performance are to adopt a character-level encoding of numbers, or to replace numbers with their corresponding scientific notation [90]. Another effective workaround consists of using more meaningful surface representations, for example, by explicitly providing the 10-base number decomposition (e.g., “832” becomes “8 100 3 10 2”) [89,91]. More advanced approaches have tried to augment the standard embeddings of number words by explicitly injecting semantic information representing quantities (for a recent survey, see [92]). One of the first attempts at learning better numeral embeddings [93] proposed to first map numbers into a log-space (to compress larger values) and train either a self-organizing map or a Gaussian mixture model with the goal of creating latent vectors encoding “number prototypes”. A similarity function is then used to project the input number into the closest prototype. Another method forced the cosine similarity between word embeddings of two numbers to reflect their actual distance on the number line [94]. Further refinements combine the prototype-based approach with the scientific notation encoding [95]. Another interesting technique exploits the Online Encyclopedia of Integer Sequences, which includes notable series of numbers that are of particular mathematical interest, to train more meaningful number embeddings [96]. The rationale was to learn number embeddings by looking at number co-occurrence across well-structured, regular number sequences, to see if this would lead to the emergence of encodings representing useful relational properties of the integers. For example, the authors discovered that one specific neuron in the embedding vector was positively activated for even numbers and negatively activated for odd numbers, suggesting the emergence of a localistic encoding of an “evenness” feature. Another recent study has also explored the adequacy of different types of encoding schemes to solve a variety of linear algebra problems over matrices, concluding that the optimal encoding might be problem dependent [97].

Overall, all these approaches for augmenting numerical embeddings allow to improve processing of small and medium-sized numbers, especially on linguistic tasks such as math word problems; however, none of them allows accurate manipulation (e.g., arithmetic) to be performed over larger quantities or numbers that are not contained in the training vocabulary.

4. Conclusions

To summarize the main findings from the recent literature, we can generally argue that although neural network models can exhibit impressive mathematical reasoning

skills, their basic abilities for representing and manipulating numbers are not yet fully developed. Even the most advanced deep learning architectures, including large language models, often fail when probed with numerical problems that are posed in a different manner than the training cases or that involve quantities well outside the training distribution. The state-of-the-art performance on challenging benchmarks such as GSM8K is steadily improving, increasing from 58% for PaLM exploiting an external calculator [98] to 78% for Minerva [50] and 92% for GPT-4 [86]. The top accuracy on ASDiv and SVAMP is around 90% for large language models with refined chain-of-thought prompting strategies [99]. However, achieving a high performance on these benchmarks might not be enough to demonstrate numerical understanding, as has also been pointed out by others [100]. Not surprisingly, indeed, the extrapolation capabilities of these gigantic neural networks are still poor: when models are not equipped with external calculators, generalization to unseen (out-of-distribution) numerical ranges requires the discovery of algorithmic procedures. This still constitutes a challenge for deep learning [101] and is indeed considered one of the frontiers in neuro-symbolic research [102]. For example, Minerva achieves over 80% calculation accuracy on 10-digit addition, but this value is never at ceiling even for easy (e.g., 5-digit) problems, and it drops to 20% on 18-digit addition, highlighting that even such a sophisticated model does not master elementary arithmetic. A similar conclusion applies to the extrapolation results presented in [83]; a finer-grained algorithmic prompting strategy allows addition problems to be consistently solved with answers of up to 19 digits in length, even if training examples were limited to 5 digits. However, the accuracy never reaches 100% and after 19 digits, the model runs out of context. The performance of deep learning models is also poor in arithmetic problems specifically designed to test compositionality [103] and it is below the human level in benchmarks that probe approximate numerical knowledge, highlighting that these models still have a limited understanding of quantities and magnitudes [38].

As others advocate, building deep learning systems that can more reliably master basic numerical concepts and simple arithmetic might be a mandatory first step towards successfully tackling complex mathematical reasoning tasks [46]. The perspective offered by decades of research in educational, developmental, and cognitive sciences could provide important insights for the design of such learning systems, especially in terms of the design of training regimens and testing procedures. For example, primary education in mathematics places a strong emphasis on the acquisition of elementary algorithms as the basis for understanding the arithmetic of natural numbers, integers, fractions, and decimals, using a variety of representation formats that exemplify the semantics of the decimal place value system [104]. Importantly, such knowledge is developed on top of the acquisition of number words, numeral systems, and counting procedures [105,106], which are already fostered during the pre-school period [107]. Furthermore, although language has a central role in this arduous learning process [108], it is not the only medium through which numerical knowledge is acquired [109]; a key role is also played by the development of basic visuo-spatial skills and geometrical intuitions [110–112], as well as by the mastery of embodied and material representational systems [63,113,114]. Grounding large language models into sensorimotor representations seems to be a promising approach to improve the basic numeracy of AI systems, and the introduction of multi-modal foundation models represent the first step in this direction [115]. Cognitive science also provides well-established operational definitions and testing procedures to assess the many facets of numerical understanding [116–118]; the adoption of similar evaluation batteries in AI research would allow to better characterize the numerical capabilities of deep learning models and more systematically identify their weaknesses [119].

Developing computational models that more faithfully simulate the acquisition of basic numerical skills would, in turn, constitute an important step toward improving the scientific understanding of our own mathematical learning abilities [120]. Deep learning has already provided key insights into the origin of our “number sense”, for example, by demonstrating that approximate numerical representations can emerge in a variety of

generative architectures [121–124] and that number acuity becomes gradually refined through unsupervised learning [125]. Although these modeling efforts have not yet been successfully extended into the realm of symbolic mathematics, the rapid progress in artificial intelligence is opening exciting prospects for bridging this gap.

As a final remark, it should also be noted that the ability of AI systems to showcase mathematical reasoning capabilities would open exciting prospects but also plunge us into significant ethical debates. Concerns predominantly revolve around the reliability and explainability of AI systems, which become increasingly more difficult to assess as these models start to be used in combination with external tools and computing software. This might become an issue, especially in educational settings, where AI can enhance learning performance and experience but also impact the learners' autonomy, introduce unwarranted biases, or amplify educational inequities [126]. As deep learning technology continues to evolve, these ethical debates should remain central to the discourse around AI and its implications for society.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The author declares no conflicts of interest.

References

- Dehaene, S. *The Number Sense: How the Mind Creates Mathematics*; Oxford University Press: New York, NY, USA, 2011.
- Nieder, A. The Neuronal Code for Number. *Nat. Rev. Neurosci.* **2016**, *17*, 366–382. <https://doi.org/10.1038/nrn.2016.40>.
- Núñez, R. Is There Really an Evolved Capacity for Number? *Trends Cogn. Sci.* **2017**, *21*, 409–424. <https://doi.org/10.1016/j.tics.2017.03.005>.
- O'Shaughnessy, D.M.; Gibson, E.; Piantadosi, S.T. The Cultural Origins of Symbolic Number. *Psychol. Rev.* **2022**, *129*, 1442–1456. <https://doi.org/10.1037/rev0000289>.
- Beller, S.; Bender, A.; Chrisomalis, S.; Jordan, F.M.; Overmann, K.A.; Saxe, G.B.; Schlimm, D. The Cultural Challenge in Mathematical Cognition. *J. Numer. Cogn.* **2018**, *4*, 448–463. <https://doi.org/10.5964/jnc.v4i2.137>.
- Newell, A.; Simon, H.A. The Logic Theory Machine: A Complex Information Processing System. *IRE Trans. Inf. Theory* **1956**, *2*, 61–79.
- Bundy, A. *The Computer Modelling of Mathematical Reasoning*; Academic Press: Cambridge, MA, USA, 1983.
- Choi, C.Q. 7 Revealing Ways AIs Fail: Neural Networks Can Be Disastrously Brittle, Forgetful, and Surprisingly Bad at Math. *IEEE Spectr.* **2021**, *58*, 42–47. <https://doi.org/10.1109/MSPEC.2021.9563958>.
- Harnad, S. The Symbol Grounding Problem. *Phys. D Nonlinear Phenom.* **1990**, *42*, 335–346. [https://doi.org/10.1016/0167-2789\(90\)90087-6](https://doi.org/10.1016/0167-2789(90)90087-6).
- Searle, J.R. Minds, Brains, and Programs. *Behav. Brain Sci.* **1980**, *3*, 417–424.
- Leibovich, T.; Ansari, D. The Symbol-Grounding Problem in Numerical Cognition: A Review of Theory, Evidence, and Outstanding Questions. *Can. J. Exp. Psychol.* **2016**, *70*, 12–23. <https://doi.org/10.1037/cep0000070>.
- Marcus, G. Deep Learning: A Critical Appraisal. *arXiv* **2018**, arXiv:1801.00631.
- Fodor, J.A.; Pylyshyn, Z.W. Connectionism and Cognitive Architecture: A Critical Analysis. *Cognition* **1988**, *28*, 3–71. [https://doi.org/10.1016/0010-0277\(88\)90031-5](https://doi.org/10.1016/0010-0277(88)90031-5).
- Cartuyvels, R.; Spinks, G.; Moens, M.-F. Discrete and Continuous Representations and Processing in Deep Learning: Looking Forward. *AI Open* **2021**, *2*, 143–159. <https://doi.org/10.1016/j.aiopen.2021.07.002>.
- Santoro, A.; Lampinen, A.; Mathewson, K.; Lillicrap, T.; Raposo, D. Symbolic Behaviour in Artificial Intelligence. *arXiv* **2021**, arXiv:2102.03406.
- Trask, A.; Hill, F.; Reed, S.; Rae, J.; Dyer, C.; Blunsom, P. Neural Arithmetic Logic Units. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018.
- Lample, G.; Charton, F. Deep Learning for Symbolic Mathematics. *arXiv* **2019**, <https://doi.org/10.48550/arXiv.1912.0141>.
- Polu, S.; Sutskever, I. Generative Language Modeling for Automated Theorem Proving. *arXiv* **2020**, arXiv:2009.03393.
- Lee, D.; Szegedy, C.; Rabe, M.N.; Loos, S.M.; Bansal, K. Mathematical Reasoning in Latent Space. *arXiv* **2019**, arXiv:1909.11851.
- Wang, M.; Deng, J. Learning to Prove Theorems by Learning to Generate Theorems. In Proceedings of the Advances in Neural Information Processing Systems; Virtual, 6–12 December 2020.
- Davies, A.; Veličković, P.; Buesing, L.; Blackwell, S.; Zheng, D.; Tomašev, N.; Tanburn, R.; Battaglia, P.; Blundell, C.; Juhász, A.; et al. Advancing Mathematics by Guiding Human Intuition with AI. *Nature* **2021**, *600*, 70–74. <https://doi.org/10.1038/s41586-021-04086-x>.
- Fawzi, A.; Balog, M.; Huang, A.; Hubert, T.; Romera-Paredes, B.; Barekatin, M.; Novikov, A.; Ruiz, F.J.; Schrittwieser, J.; Swirszcz, G.; et al. Discovering Faster Matrix Multiplication Algorithms with Reinforcement Learning. *Nature* **2022**, *610*, 47–53. <https://doi.org/10.1038/s41586-022-05172-4>.

23. Drori, I.; Zhang, S.; Shuttleworth, R.; Tang, L.; Lu, A.; Ke, E.; Liu, K.; Chen, L.; Tran, S.; Cheng, N.; et al. A Neural Network Solves, Explains, and Generates University Math Problems by Program Synthesis and Few-Shot Learning at Human Level. *Proc. Natl. Acad. Sci. USA* **2022**, *119*, e2123433119. <https://doi.org/10.1073/pnas.2123433119>.
24. Davis, E. Limits of an AI Program for Solving College Math Problems. *arXiv* **2022**, arXiv:2208.06906.
25. Davis, E. The Use of Deep Learning for Symbolic Integration: A Review of (Lample and Charton, 2019). *arXiv* **2019**, arXiv:1912.05752.
26. Claessens, A.; Engel, M. How Important Is Where You Start? Early Mathematics Knowledge and Later School Success. *Teach. Coll. Rec.* **2013**, *115*, 1–29. <https://doi.org/10.1177/016146811311500603>.
27. Nguyen, T.; Watts, T.W.; Duncan, G.J.; Clements, D.H.; Sarama, J.S.; Wolfe, C.; Spitzer, M.E. Which Preschool Mathematics Competencies Are Most Predictive of Fifth Grade Achievement? *Early Child. Res. Q.* **2016**, *36*, 550–560. <https://doi.org/10.1016/j.ecresq.2016.02.003>.
28. Jordan, N.C.; Kaplan, D.; Ramineni, C.; Locuniak, M.N. Early Math Matters: Kindergarten Number Competence and Later Mathematics Outcomes. *Dev. Psychol.* **2009**, *45*, 850–867. <https://doi.org/10.1037/a0014939>.
29. Faldu, K.; Sheth, A.; Kikani, P.; Gaur, M.; Avasthi, A. Towards Tractable Mathematical Reasoning: Challenges, Strategies, and Opportunities for Solving Math Word Problems. *arXiv* **2021**, arXiv:2111.05364.
30. Huang, D.; Shi, S.; Lin, C.-Y.; Yin, J.; Ma, W.-Y. How Well Do Computers Solve Math Word Problems? Large-Scale Dataset Construction and Evaluation. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; Association for Computational Linguistics: Stroudsburg, PA, USA, 2016; Volume 2, pp. 887–896.
31. Wang, Y.; Liu, X.; Shi, S. Deep Neural Solver for Math Word Problems. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Copenhagen, Denmark, 9–11 September 2017.
32. Ling, W.; Yogatama, D.; Dyer, C.; Blunsom, P. Program Induction by Rationale Generation: Learning to Solve and Explain Algebraic Word Problems. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; Volume 1, pp. 158–167.
33. Amini, A.; Gabriel, S.; Lin, S.; Koncel-Kedziorski, R.; Choi, Y.; Hajishirzi, H. MathQA: Towards Interpretable Math Word Problem Solving with Operation-Based Formalisms. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Minneapolis, MN, USA, 2–7 June 2019; Volume 1, pp. 2357–2367.
34. Miao, S.; Liang, C.-C.; Su, K.-Y. A Diverse Corpus for Evaluating and Developing English Math Word Problem Solvers. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Virtual, 5–10 July 2020; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 975–984.
35. Patel, A.; Bhattamishra, S.; Goyal, N. Are NLP Models Really Able to Solve Simple Math Word Problems? In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Virtual Event, 6–11 June; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 2080–2094.
36. Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. Training Verifiers to Solve Math Word Problems. *arXiv* **2021**, arXiv:2110.14168.
37. Gupta, H.; Varshney, N.; Mishra, S.; Pal, K.K.; Sawant, S.A.; Scaria, K.; Goyal, S.; Baral, C. “John Is 50 Years Old, Can His Son Be 65?” Evaluating NLP Models’ Understanding of Feasibility. In Proceedings of the The 17th Conference of the European Chapter of the Association for Computational Linguistics, Dubrovnik, Croatia, 2–6 May 2023.
38. Fujisawa, I.; Kanai, R. Logical Tasks for Measuring Extrapolation and Rule Comprehension. *arXiv* **2022**, arXiv:2211.07727.
39. Cognolato, S.; Testolin, A. Transformers Discover an Elementary Calculation System Exploiting Local Attention and Grid-like Problem Representation. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), Padua, Italy, 18–23 July 2022.
40. Madsen, A.; Johansen, A.R. Neural Arithmetic Units. In Proceedings of the International Conference on Learning Representations; Virtual Conference, 25–30 April 2020.
41. Anil, C.; Wu, Y.; Andreassen, A.; Lewkowycz, A.; Misra, V.; Ramasesh, V.; Slone, A.; Gur-Ari, G.; Dyer, E.; Neyshabur, B. Exploring Length Generalization in Large Language Models. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 28 November–9 December 2022.
42. Saxton, D.; Grefenstette, E.; Hill, F.; Kohli, P. Analysing Mathematical Reasoning Abilities of Neural Models. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
43. Dua, D.; Wang, Y.; Dasigi, P.; Stanovsky, G.; Singh, S.; Gardner, M. Drop: A Reading Comprehension Benchmark Requiring Discrete Reasoning over Paragraphs. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; pp. 2368–2378.
44. Lin, B.Y.; Lee, S.; Khanna, R.; Ren, X. Birds Have Four Legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Virtual Conference, 16–20 November 2020; pp. 6862–6868.
45. Mishra, S.; Mitra, A.; Varshney, N.; Sachdeva, B.; Clark, P.; Baral, C.; Kalyan, A. NumGLUE: A Suite of Fundamental yet Challenging Mathematical Reasoning Tasks. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Dublin, Ireland, 22–27 May 2022; pp. 3505–3523.
46. Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; Steinhardt, J. Measuring Mathematical Problem

- Solving With the MATH Dataset. In Proceedings of the Advances in Neural Information Processing Systems, Virtual Conference, 6–14 December 2021.
47. Mishra, S.; Finlayson, M.; Lu, P.; Tang, L.; Welleck, S.; Baral, C.; Rajpurohit, T.; Tafjord, O.; Sabharwal, A.; Clark, P.; et al. Lila: A Unified Benchmark for Mathematical Reasoning. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 7–11 December 2022; pp. 5807–5832.
 48. Lu, P.; Bansal, H.; Xia, T.; Liu, J.; Li, C.; Hajishirzi, H.; Cheng, H.; Chang, K.; Galley, M.; Gao, J. MathVista: Evaluating Math Reasoning in Visual Contexts with GPT-4V, Bard, and Other Large Multimodal Models. *arXiv* **2023**, arXiv.2310.02255.
 49. Lewkowycz, A.; Andreassen, A.; Dohan, D.; Dyer, E.; Michalewski, H.; Ramasesh, V.; Slone, A.; Anil, C.; Schlag, I.; Gutman-Solo, T.; et al. Solving Quantitative Reasoning Problems with Language Models. *arXiv* **2022**, arXiv.2206.14858.
 50. Frieder, S.; Pinchetti, L.; Griffiths, R.-R.; Salvatori, T.; Lukasiewicz, T.; Petersen, P.C.; Chevalier, A.; Berner, J. Mathematical Capabilities of ChatGPT. *arXiv* **2023**, <https://doi.org/10.48550/arXiv.2301.13867>.
 51. Xie, Z.; Sun, S. A Goal-Driven Tree-Structured Neural Model for Math Word Problems. In Proceedings of the International Joint Conference on Artificial Intelligence, Macao, China, 10–16 August 2019; pp. 5299–5305.
 52. Wang, L.; Wang, Y.; Cai, D.; Zhang, D.; Liu, X. Translating a Math Word Problem to a Expression Tree. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Brussels, Belgium, 31 October–4 November 2018; pp. 1064–1069.
 53. Zhang, J.; Wang, L.; Lee, R.K.-W.; Bin, Y.; Wang, Y.; Shao, J.; Lim, E.-P. Graph-to-Tree Learning for Solving Math Word Problems. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Virtual Event, 5–10 July 2020, Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 3928–3937.
 54. Ran, Q.; Lin, Y.; Li, P.; Zhou, J.; Liu, Z. NumNet: Machine Reading Comprehension with Numerical Reasoning. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Hong Kong, China, 3–7 November 2019; pp. 2474–2484.
 55. Csordás, R.; Irie, K.; Schmidhuber, J. The Neural Data Router: Adaptive Control Flow in Transformers Improves Systematic Generalization. In Proceedings of the International Conference on Learning Representations, Virtual Conference, 25–29 April 2022.
 56. Petruzzellis, F.; Testolin, A.; Sperduti, A. A Hybrid System for Systematic Generalization in Simple Arithmetic Problems. In Proceedings of the 17th International Workshop on Neural-Symbolic Learning and Reasoning, Siena, Italy, 3–5 July 2023; Volume 3432, pp. 289–301.
 57. Jelassi, S.; D’Ascoli, S.; Domingo-Enrich, C.; Wu, Y.; Li, Y.; Charton, F. Length Generalization in Arithmetic Transformers. *arXiv* **2023**, <https://doi.org/10.48550/arXiv.2306.15400>.
 58. Santoro, A.; Bartunov, S.; Botvinick, M.; Wierstra, D.; Lillicrap, T. Meta-Learning with Memory-Augmented Neural Networks. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; Volume 4, pp. 1842–1850.
 59. Graves, A.; Wayne, G.; Reynolds, M.; Harley, T.; Danihelka, I.; Grabska-Barwińska, A.; Gómez Colmenarejo, S.; Grefenstette, E.; Ramalho, T.; Agapiou, J.; et al. Hybrid Computing Using a Neural Network with Dynamic External Memory. *Nature* **2016**, *538*, 471–476. <https://doi.org/10.1038/nature20101>.
 60. Kaiser, Ł.; Sutskever, I. Neural GPUs Learn Algorithms. In Proceedings of the International Conference on Learning Representations; San Diego, CA, USA, 7–9 May 2015.
 61. Kim, S.; Nam, H.; Kim, J.; Jung, K. Neural Sequence-to-Grid Module for Learning Symbolic Rules. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021.
 62. Overmann, K.A. The Role of Materiality in Numerical Cognition. *Quat. Int.* **2016**, *405*, 42–51. <https://doi.org/10.1016/j.quaint.2015.05.026>.
 63. d’Errico, F.; Doyon, L.; Colagè, I.; Queffelec, A.; Le Vraux, E.; Giacobini, G.; Vandermeersch, B.; Maureille, B. From Number Sense to Number Symbols. An Archaeological Perspective. *Philos. Trans. R. Soc. B Biol. Sci.* **2018**, *373*, 20160518. <https://doi.org/10.1098/rstb.2016.0518>.
 64. Petruzzellis, F.; Chen, L.X.; Testolin, A. Learning to Solve Arithmetic Problems with a Virtual Abacus. In Proceedings of the Northern Lights Deep Learning Workshop, Tromsø, Norway, 10–12 January 2023.
 65. Sabathiel, S.; Solstad, T.; Testolin, A.; Petruzzellis, F. Self-Communicating Deep Reinforcement Learning Agents Develop External Number Representations. In Proceedings of the Northern Lights Deep Learning Workshop, Tromsø, Norway, 10–12 January 2022.
 66. Parisi, A.; Zhao, Y.; Fiedel, N. TALM: Tool Augmented Language Models. *arXiv* **2022**, <https://doi.org/10.48550/arXiv.2205.12255>.
 67. Karpas, E.; Abend, O.; Belinkov, Y.; Lenz, B.; Lieber, O.; Ratner, N.; Shoham, Y.; Bata, H.; Levine, Y.; Muhlgay, D.; et al. MRKL Systems: A Modular, Neuro-Symbolic Architecture That Combines Large Language Models, External Knowledge Sources and Discrete Reasoning. *arXiv* **2022**, <https://doi.org/10.48550/arXiv.2205.00445>.
 68. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models Are Few-Shot Learners. In Proceedings of the Advances in Neural Information Processing Systems, Virtual Conference, 6–12 December 2020.
 69. Henighan, T.; Kaplan, J.; Katz, M.; Chen, M.; Hesse, C.; Jackson, J.; Jun, H.; Brown, T.B.; Dhariwal, P.; Gray, S.; et al. Scaling Laws for Autoregressive Generative Modeling. *arXiv* **2020**, arXiv.2010.14701.
 70. Rae, J.W.; Borgeaud, S.; Cai, T.; Millican, K.; Hoffmann, J.; Song, F.; Aslanides, J.; Henderson, S.; Ring, R.; Young, S.; et al. Scaling Language Models: Methods, Analysis & Insights from Training Gopher. *arXiv* **2021**, arXiv.2112.11446.
 71. Pal, K.K.; Baral, C. Investigating Numeracy Learning Ability of a Text-to-Text Transfer Model. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Virtual Conference, 7–11 November 2021; pp. 3095–3101.

72. Razeghi, Y.; Logan, R.L.; Gardner, M.; Singh, S. Impact of Pretraining Term Frequencies on Few-Shot Reasoning. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Abu Dhabi, United Arab Emirates, 7–11 December 2022; pp. 840–854.
73. Geva, M.; Gupta, A.; Berant, J. Injecting Numerical Reasoning Skills into Language Models. In Proceedings of the The 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 946–958.
74. Yang, P.-J.; Chen, Y.T.; Chen, Y.; Cer, D. NT5?! Training T5 to Perform Numerical Reasoning. *arXiv* **2021**, arXiv:2104.07307.
75. Luo, H.; Sun, Q.; Xu, C.; Zhao, P.; Lou, J.; Tao, C.; Geng, X.; Lin, Q.; Chen, S.; Zhang, D. WizardMath: Empowering Mathematical Reasoning for Large Language Models via Reinforced Evol-Instruct. *arXiv* **2023**, <https://doi.org/10.48550/arXiv.2308.09583>.
76. Azerbayev, Z.; Schoelkopf, H.; Paster, K.; Santos, M.D.; McAleer, S.; Jiang, A.Q.; Deng, J.; Biderman, S.; Welleck, S. Llemma: An Open Language Model For Mathematics. *arXiv* **2023**, arXiv:2310.10631.
77. Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; de Oliveira Pinto, H.P.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. Evaluating Large Language Models Trained on Code. *arXiv* **2021**, arXiv:2107.03374.
78. Nye, M.; Andreassen, A.J.; Gur-Ari, G.; Michalewski, H.; Austin, J.; Bieber, D.; Dohan, D.; Lewkowycz, A.; Bosma, M.; Luan, D.; et al. Show Your Work: Scratchpads for Intermediate Computation with Language Models. In Proceedings of the International Conference on Learning Representations, Virtual, 3–7 May, 2021.
79. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 28 November–9 December 2022.
80. Banino, A.; Balaguer, J.; Blundell, C. PonderNet: Learning to Ponder. In Proceedings of the International Conference on Machine Learning: Workshop on Automated Machine Learning, Virtual Conference, 18–24 July 2021.
81. Graves, A. Adaptive Computation Time for Recurrent Neural Networks. *arXiv* **2016**, arXiv:1603.08983.
82. Zhou, H.; Nova, A.; Larochelle, H.; Courville, A.; Neyshabur, B.; Sedghi, H. Teaching Algorithmic Reasoning via In-Context Learning. *arXiv* **2022**, <https://doi.org/10.48550/arXiv.2211.09066>.
83. Imani, S.; Du, L.; Shrivastava, H. MathPrompter: Mathematical Reasoning Using Large Language Models. *arXiv* **2023**, arXiv:2303.0539.
84. Petruzzellis, F.; Testolin, A.; Sperduti, A. Benchmarking GPT-4 on Algorithmic Problems: A Systematic Evaluation of Prompting Strategies. *under review*.
85. OpenAI GPT-4 Technical Report. *arXiv* **2023**, arXiv:2303.08774.
86. Wallace, E.; Wang, Y.; Li, S.; Singh, S.; Gardner, M. Do NLP Models Know Numbers? Probing Numeracy in Embeddings. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Hong Kong, China, 3–7 November 2019; pp. 5307–5315.
87. Naik, A.; Ravichander, A.; Rose, C.; Hovy, E. Exploring Numeracy in Word Embeddings. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2019; pp. 3374–3380.
88. Nogueira, R.; Jiang, Z.; Lin, J. Investigating the Limitations of Transformers with Simple Arithmetic Tasks. *arXiv* **2021**, arXiv:2102.13019.
89. Zhang, X.; Ramachandran, D.; Tenney, I.; Elazar, Y.; Roth, D. Do Language Embeddings Capture Scales? In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Virtual Conference, 16–20 November 2020; pp. 4889–4896.
90. Kim, J.; Hong, G.; Kim, K.M.; Kang, J.; Myaeng, S.H. Have You Seen That Number? Investigating Extrapolation in Question Answering Models. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Virtual, 7–11 November 2021; pp. 7031–7037.
91. Thawani, A.; Pujara, J.; Ilievski, F.; Szekely, P. Representing Numbers in NLP: A Survey and a Vision. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Virtual Event, 2–7 June 2021; pp. 644–656.
92. Jiang, C.; Nian, Z.; Guo, K.; Chu, S.; Zhao, Y.; Shen, L.; Tu, K. Learning Numeral Embedding. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Virtual, 16–20 November 2020; pp. 2586–2599.
93. Sundararaman, D.; Si, S.; Subramanian, V.; Wang, G.; Hazarika, D.; Carin, L. Methods for Numeracy-Preserving Word Embeddings. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Virtual, 16–20 November 2020; pp. 4742–4753.
94. Jin, Z.; Jiang, X.; Wang, X.; Liu, Q.; Wang, Y.; Ren, X.; Qu, H. NumGPT: Improving Numeracy Ability of Generative Pre-Trained Models. *arXiv* **2021**, <https://doi.org/10.48550/arXiv.2109.03137>.
95. Ryskina, M.; Knight, K. Learning Mathematical Properties of Integers. In Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, Punta Cana, Dominican Republic, 11 November 2021; pp. 389–395.
96. Charton, F. Linear Algebra with Transformers. *arXiv* **2021**, arXiv:2112.01898.
97. Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H.W.; Sutton, C.; Gehrmann, S.; et al. PaLM: Scaling Language Modeling with Pathways. *arXiv* **2022**, arXiv:2204.02311.
98. Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; Zhou, D. Self-Consistency Improves Chain of Thought Reasoning in Language Models. *arXiv* **2022**, arXiv:2203.11171.
99. Davis, E. Mathematics, Word Problems, Common Sense, and Artificial Intelligence. *arXiv* **2023**, arXiv:2301.09723.
100. Welleck, S.; West, P.; Cao, J.; Choi, Y. Symbolic Brittleness in Sequence Models: On Systematic Generalization in Symbolic

- Mathematics. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 22 February–1 March 2022; Volume 36, pp. 8629–8637.
101. Hitzler, P.; Sarker, M.K. (Eds.) *Neuro-Symbolic Artificial Intelligence: The State of the Art*; Frontiers in Artificial Intelligence and Applications; IOS Press: Clifton, VA, USA, 2021; Volume 342; ISBN 9781643682440.
 102. Kudo, K.; Aoki, Y.; Kuribayashi, T.; Brassard, A.; Yoshikawa, M.; Sakaguchi, K.; Inui, K. Do Deep Neural Networks Capture Compositionality in Arithmetic Reasoning? *arXiv* **2023**, arXiv.2302.07866.
 103. Sarama, J.; Clements, D.H. *Early Childhood Mathematics Education Research*; Routledge: Abingdon, UK, 2009; ISBN 9780203883785.
 104. Carey, S.; Barner, D. Ontogenetic Origins of Human Integer Representations. *Trends Cogn. Sci.* **2019**, *23*, 823–835. <https://doi.org/10.1016/j.tics.2019.07.004>.
 105. Rittle-Johnson, B.; Siegler, R.S.; Alibali, M.W. Developing Conceptual Understanding and Procedural Skill in Mathematics: An Iterative Process. *J. Educ. Psychol.* **2001**, *93*, 346–362. <https://doi.org/10.1037/0022-0663.93.2.346>.
 106. Anders, Y.; Rossbach, H.; Weinert, S.; Ebert, S.; Kuger, S.; Lehl, S.; Maurice, J. Von Home and Preschool Learning Environments and Their Relations to the Development of Early Numeracy Skills. *Early Child. Res. Q.* **2012**, *27*, 231–244. <https://doi.org/10.1016/j.ecresq.2011.08.003>.
 107. Turan, E.; De Smedt, B. Understanding Mathematical Development in Preschool Children: The Association between Mathematical Language and Mathematical Abilities. *Cogn. Dev.* **2023**, *66*, 101318. <https://doi.org/10.1016/j.cogdev.2023.101318>.
 108. Gelman, R.; Butterworth, B. Number and Language: How Are They Related? *Trends Cogn. Sci.* **2005**, *9*, 6–10. <https://doi.org/10.1016/j.tics.2004.11.004>.
 109. Kellman, P.J.; Massey, C.M.; Son, J.Y. Perceptual Learning Modules in Mathematics: Enhancing Students' Pattern Recognition, Structure Extraction, and Fluency. *Top. Cogn. Sci.* **2010**, *2*, 285–305. <https://doi.org/10.1111/j.1756-8765.2009.01053.x>.
 110. Piazza, M. Neurocognitive Start-up Tools for Symbolic Number Representations. *Trends Cogn. Sci.* **2010**, *14*, 542–551. <https://doi.org/10.1016/j.tics.2010.09.008>.
 111. Dehaene, S. Origins of Mathematical Intuitions: The Case of Arithmetic. *Ann. N. Y. Acad. Sci.* **2009**, *1156*, 232–259. <https://doi.org/10.1111/j.1749-6632.2009.04469.x>.
 112. Bender, A.; Beller, S. Nature and Culture of Finger Counting: Diversity and Representational Effects of an Embodied Cognitive Tool. *Cognition* **2012**, *124*, 156–182. <https://doi.org/10.1016/j.cognition.2012.05.005>.
 113. Lakoff, G.; Núñez, R. *Where Mathematics Comes from: How the Embodied Mind Brings Mathematics into Being*; Basic Books: New York, NY, USA, 2000.
 114. Yang, Z.; Li, L.; Lin, K.; Wang, J.; Lin, C.-C.; Liu, Z.; Wang, L. The Dawn of LMMs: Preliminary Explorations with GPT-4V(Ision). *arXiv* **2023**, arXiv.2309.17421.
 115. Delazer, M.; Girelli, L.; Granà, A.; Domahs, F. Number Processing and Calculation—Normative Data from Healthy Adults. *Clin. Neuropsychol.* **2003**, *17*, 331–350. <https://doi.org/10.1076/clin.17.3.331.18092>.
 116. Clements, D.H.; Sarama, J.H.; Liu, X.H. Development of a Measure of Early Mathematics Achievement Using the Rasch Model: The Research-Based Early Maths Assessment. *Educ. Psychol.* **2008**, *28*, 457–482. <https://doi.org/10.1080/01443410701777272>.
 117. Purpura, D.J.; Lonigan, C.J. Early Numeracy Assessment: The Development of the Preschool Early Numeracy Scales. *Early Educ. Dev.* **2015**, *26*, 286–313. <https://doi.org/10.1080/10409289.2015.991084>.
 118. Testolin, A.; Hou, K.; Zorzi, M. Large-Scale Generative AI Models Lack Visual Number Sense. *under review*.
 119. Testolin, A. The Challenge of Modeling the Acquisition of Mathematical Concepts. *Front. Hum. Neurosci.* **2020**, *14*, 100. <https://doi.org/10.3389/fnhum.2020.00100>.
 120. Boccato, T.; Testolin, A.; Zorzi, M. Learning Numerosity Representations with Transformers: Number Generation Tasks and Out-of-Distribution Generalization. *Entropy* **2021**, *23*, 857.
 121. Testolin, A.; Dolfi, S.; Rochus, M.; Zorzi, M. Visual Sense of Number vs. Sense of Magnitude in Humans and Machines. *Sci. Rep.* **2020**, *10*, 10045. <https://doi.org/10.1038/s41598-020-66838-5>.
 122. Stoianov, I.; Zorzi, M. Emergence of a “visual Number Sense” in Hierarchical Generative Models. *Nat. Neurosci.* **2012**, *15*, 194–196. <https://doi.org/10.1038/nn.2996>.
 123. Zhao, S.; Ren, H.; Yuan, A.; Song, J.; Goodman, N.; Ermon, S. Bias and Generalization in Deep Generative Models: An Empirical Study. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, Canada, 3–8 December 2018.
 124. Testolin, A.; Zou, W.Y.; McClelland, J.L. Numerosity Discrimination in Deep Neural Networks: Initial Competence, Developmental Refinement and Experience Statistics. *Dev. Sci.* **2020**, *23*, e12940. <https://doi.org/10.1111/desc.12940>.
 125. Nguyen, A.; Ngo, H.N.; Hong, Y.; Dang, B.; Nguyen, B.P.T. Ethical Principles for Artificial Intelligence in Education. *Educ. Inf. Technol.* **2023**, *28*, 4221–4241. <https://doi.org/10.1007/s10639-022-11316-w>.
 126. Lu, P.; Qiu, L.; Yu, W.; Welleck, S.; Chang, K. A Survey of Deep Learning for Mathematical Reasoning. *arXiv* **2022**, arXiv.2212.10535.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.