CURRICULUM LEARNING SHAPES THE COMPUTATION LEARNED BY ARITHMETIC TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Do transformers learn arithmetic by mirroring digit-wise algorithms, or by exploiting statistical shortcuts? Here we study how an encoder transformer learns to solve addition and find that the most significant digits are learned before the less significant ones, in contrast to how one would compute addition by hand. We exactly solve the learning dynamics of a linear model on the same task and show that it learns a continuous magnitude representation of the sum, despite the task being encoded symbolically. As this representation is learned, it encodes the most significant digits accurately first, since errors in this estimate disproportionately affect less significant digits. Next we apply curriculum learning to the transformer to force it to learn the task in the canonical order, and find that this shifts the model from an estimation-like strategy to digit-wise computation, and improves generalisation performance to tasks with a similar computational structure as addition. These findings demonstrate that different strategies compete inside neural networks, and structuring the training curriculum can fundamentally alter a model's internal strategy.

1 INTRODUCTION

Many current foundation models are based on transformers, yet the algorithms learned by these models remain poorly understood (Vaswani et al., 2017; Bommasani et al., 2021). Final accuracy alone cannot distinguish a robust algorithm from a solution that exploits correlations specific to the training distribution. Such solutions can perform similarly in distribution while differing under transfer or distribution shift (Geirhos et al., 2020; D'Amour et al., 2022).

To shed light on the internal computations of the model, several interpretability methods have been developed based on representation probing and causal analysis (Belinkov, 2022; Meng et al., 2022; Wang et al., 2023; Geiger et al., 2025). A central difficulty here is validation, since for natural tasks, the computation implemented by the model is unknown, making it difficult to determine whether a proposed explanation is faithful or complete. Controlled benchmarks solve this by providing a ground truth, such as a known underlying mechanism or a precisely specified task, against which a candidate explanation can be directly tested (Lindner et al., 2023; Chughtai et al., 2023).

One such benchmark is arithmetic, where inputs, outputs, intermediate variables such as carries, and the training distribution can all be controlled easily. This setting has been used to study how data representation and supervision affect generalisation (Lee et al., 2024; Cho et al., 2024; McLeish et al., 2024), follow the formation of algorithms solving for modular addition (Nanda et al., 2023), and discover carry-related computations in full addition (Quirke et al., 2024). More recently, Zhao et al. (2026) showed that autoregressive transformers learn addition from most to least significant digit and related this order to correlations in the training data. It remains unclear how these correlations give rise to the observed acquisition order and shape the learned representation, whether this order can be deliberately redirected, and how such interventions affect the properties of the solution.

Cognitive science provides a complementary perspective to arithmetic acquisition. Children develop several strategies for the same operation, which coexist and change through experience and adaptation to the problem rather than being replaced by a single transition (Siegler & Shrager, 1984; Lemaire & Siegler, 1995; Siegler, 1996). This literature studies trial-level errors and learning trajectories because aggregate accuracy can hide changes in strategy use (Siegler, 1987; Shrager & Siegler, 1998). It

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

therefore suggests examining acquisition order, error profiles, and the effects of curriculum in neural networks. We use this to motivate questions and analyses:

> *What determines the order in which a transformer acquires arithmetic subskills, and can curriculum change the solution that is ultimately learned?*

We address this question by first reproducing the recently observed most-significant digit first acquisition of addition (Zhao et al., 2026) in an encoder transformer, showing that this order does not require autoregressive decoding. **Our main contributions** are:

- We derive an exact learning solution of a tractable linear model on addition, and find it learns a magnitude representation from which an ideal decoder resolves digits in order of most to least significant (Section 3).
- We use target interventions, behavioural tests, and mechanistic analysis to distinguish estimation-like and digit-wise computational signatures in a non-autoregressive transformer (Section 4).
- We show that a length curriculum which reverses acquisition order, produces carry-aligned computations, and improves generalisation and adaptation to similar tasks (Section 5).

1.1 Related Works

Interpreting mathematical computation in transformers. Mechanistic studies have identified distinct algorithms for modular addition (Nanda et al., 2023; Zhong et al., 2023), while for full addition, carry-related computations have been discovered in small transformers (Quirke & Barez, 2024; Kruthoff, 2024), and numerical representations and arithmetic heuristics in pretrained language models (Zhou et al., 2024; Nikankin et al., 2025). Experiments also show that data representation and supervision strongly affect arithmetic generalisation (Lee et al., 2024; Cho et al., 2024; McLeish et al., 2024). Here we link these mechanistic accounts to the dynamics of acquisition in order to clarify why optimisation favours particular solutions and reveal their limitations.

Learning dynamics and curriculum. A neural network's generalisation and convergence depend heavily on its training trajectory, where early experience can have lasting effects and optimisation can get stuck in suboptimal solutions (Achille et al., 2019; Saxe et al., 2013; Sarao Mannelli et al., 2024; Berthier et al., 2025). Curriculum learning provides a way to influence these dynamics towards a desired solution (Bengio et al., 2009; Wu et al., 2021), though identifying an effective curriculum remains task- and model-dependent. Theory results have shown that curricula can accelerate learning (Weinshall et al., 2018; Saglietti et al., 2022), steer optimisation towards better solutions (Sarao Mannelli et al., 2024), reduce the cost of learning parity functions (Cornacchia & Mossel, 2023), and enable complex feature acquisition through a learnable staircase of simpler ones (Abbe et al., 2021). Despite this, empirical comparisons have found only limited benefits on standard deep learning benchmarks (Wu et al., 2021). Here we show that for transformer addition a human-interpretable curriculum accelerates learning and improves generalisation and transfer.

Arithmetic strategies in cognitive development. Children develop several arithmetic strategies, including counting all items, counting on from an addend, retrieving known facts, and decomposing problems into simpler ones (Siegler & Shrager, 1984; Siegler, 1987). Strategy-choice models and the overlapping-waves framework describe these strategies as coexisting, with their use changing through experience and adaptation to the problem (Siegler & Shrager, 1984; Siegler, 1996), while strategy discovery models explain how children develop more efficient procedures (Shrager & Siegler, 1998). Beyond exact calculation, children use rounding or truncation to estimate symbolic sums (Lemaire et al., 2000), and can make approximate arithmetic judgments even before mastering the exact calculations (Gilmore et al., 2007). Additionally, errors contain information about the underlying procedure, as systematic mistakes can reflect an incorrect rule rather than an imprecise estimate (Brown & Burton, 1978). Here, we use *estimation-like* and *digit-wise* to describe model behaviour and representations for corresponding computational signatures, without assuming that the models reproduce human strategies.

2 MOST-SIGNIFICANT-DIGIT-FIRST ACQUISITION

We train a transformer to learn addition. Each datapoint is a sequence containing two n-digit base-10 operands with a plus token in between. $n+1$ answer slots are appended where the model must output digits corresponding to the sum (figure 1a). We use an encoder-only transformer, so all answer digits are predicted in parallel without conditioning on preceding answer digits. Digits and the plus token are represented as categorical tokens, by encoding them with one-hot vectors, ensuring no explicit numerical-value features are present. The model has four layers, four attention heads per layer, embedding dimension 256, and feedforward dimension 2048. We optimise the average cross-entropy loss over all answer slots using AdamW with learning rate 5×10^{-5}, weight decay 0.01, and batch size 64. Data is generated online at every training step, to avoid the model memorising a fixed dataset.

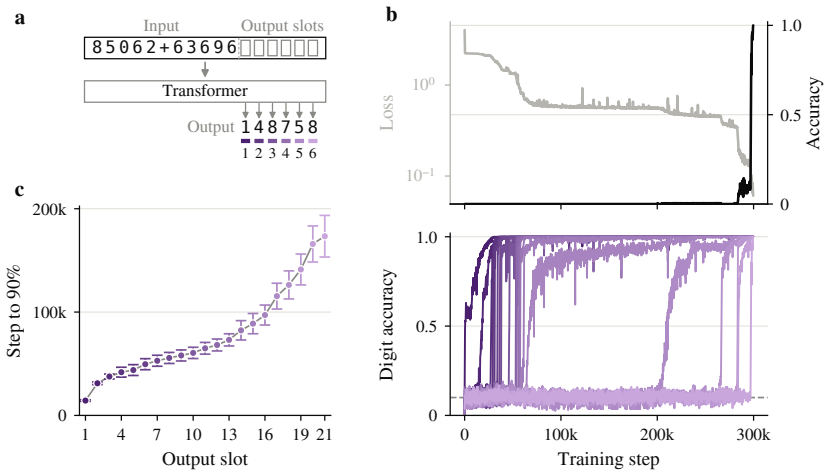


Figure 1: **Transformers acquire addition from most to least significant digit.** **(a)** Schematic illustration of the task. The model is fed a sequence containing two input numbers and has to predict the sum's digits in parallel. **(b)** Top: loss and exact-match accuracy during training. Bottom: digit-wise accuracy at different output positions. **(c)** Number of training steps for each digit to reach 90% accuracy, averaged over 10 seeds. Despite the model computing in parallel, digits are learned from left (most significant) to right (least significant).

Digits are learned backwards. The top plot in figure 1b shows an apparently abrupt transition in exact-match accuracy, along with successive drops in the loss. Inspecting the digit-wise accuracy shows that this transition corresponds to a sequence of learning stages (figure 1b bottom). The model first becomes accurate on the most significant output digit and then acquires progressively less significant positions. While the exact timing varies across random seeds, the order remains stable (figure 1c and Section C.1). Zhao et al. (2026) observed a similar reverse-order digit learning in transformers trained autoregressively. Here each answer digit is predicted without conditioning on preceding answer digits, showing that this order does not require the autoregressive left-to-right output conditioning. The resulting digit learning order is opposite to the standard addition algorithm, which resolves positions from least to most significant as carries propagate leftward. This does not exclude the possibility that the model represents carries internally, but it shows that the observable output subskills are not acquired in carry-propagation order. The question is therefore why parallel training makes the most significant digit accessible first. We next analyse how the statistics of addition favour this order in a tractable model of the learning dynamics.

3 LINEAR NETWORKS LEARN AN ESTIMATE OF THE MAGNITUDE ON ADDITION

We investigate a contrast between *estimation-like processing*, which prioritises the overall magnitude of the result, and *digit-wise computation*, which combines operations on individual digits with carry

information. This distinction is motivated by work on computational estimation (Lemaire et al., 2000; Ganor-Stern, 2018) and decomposed processing in human arithmetic (Klein et al., 2010).

Setup. Let us consider the task of numerical addition of numbers represented in some base $b \geq 2$, with a fixed number of input digits $n \geq 1$. The two inputs a, a' are assumed to be drawn independently and uniformly on $\{0, \ldots, b^n - 1\}$. We number digits from the most significant, so the i-th input digit has place value b^{n-i}. The numerical value of the answer and its $n+1$ output digits take the form

$$S = a + a', \qquad y_i = \left\lfloor S / b^{n-i} \right\rfloor \bmod b, \qquad i = 0, \ldots, n, \tag{1}$$

where $y_0 = \mathbf{1}\{S \geq b^n\}$ is the overflow digit. The inputs $x \in \mathbb{R}^{2nb}$ and targets $y \in \mathbb{R}^{1+nb}$ are centered one-hot encodings of the input digits and of $(y_0, \ldots, y_n)$ (Section B.1).

Linear network. To understand what kind of strategy a neural network might employ early in training on addition, we study a two layer linear network with a single hidden unit. Despite the limited expressiveness, linear networks display complex yet solvable learning dynamics (Saxe et al., 2013). The network is defined as $\hat{y} = W_2 W_1 x$ with $W_1 \in \mathbb{R}^{1 \times 2nb}$, $W_2 \in \mathbb{R}^{(1+nb) \times 1}$ with hidden unit given by $h(x) = W_1 x$. We train with gradient flow on the mean squared error $\frac{1}{2}\,\mathbb{E}\,\|\hat{y} - y\|^2$.

The network learns a magnitude representation. Using the approach from (Saxe et al., 2013), we can determine the structure that the model learns by taking a singular value decomposition of the input-output correlation matrix $\mathbb{E}[yx^\top]$. It turns out that every digit of the sum is linearly uncorrelated with every input digit, except for the overflow digit, whose correlation with each input digit is proportional to that digit's place value b^{n-i}. This results in $\mathbb{E}[yx^\top]$ having rank one, with a singular value s_n and corresponding right singular vector v satisfying $v^\top x = (S - \mathbb{E}\,S)/\operatorname{sd}(S)$ (Section B.1). For any generic initialisation satisfying balanced weights, the gradient flow equation becomes solvable and we find that the hidden unit converges, up to a sign set by the initialisation, to the standardised sum:

$$\boxed{h(x) \;\longrightarrow\; \sqrt{s_n}\,\frac{S - \mathbb{E}\,S}{\operatorname{sd}(S)}, \qquad s_n = \sqrt{\tfrac{1 - b^{-2n}}{6}}} \tag{2}$$

(Sections B.2 and B.3). Notably, the overflow digit plays a crucial role here, and the magnitude representation is not learned when it is absent.

A perfect decoder on the magnitude representation learns significant digits first. The linear model is incapable of solving the task, as the loss does not reach zero in the infinite time limit (Section B.2). To understand how a more capable model would learn the digits using this representation, we apply a perfect decoder to the hidden unit.

The hidden unit during training takes the form:

$$h = w_{1\parallel}\,\frac{S - \mathbb{E}\,S}{\operatorname{sd}(S)} + w_{1\perp}\,\eta, \qquad w_{1\parallel} = W_1 v,$$

where $w_{1\perp}\,\eta$ is the contribution of the component of W_1 orthogonal to v, with η a unit-variance variable uncorrelated with S (Section B.4). The first term is a rescaled copy of S, which determines every output digit exactly, so a decoder can invert it to recover an estimate $\hat{S}$ of the sum and read each digit off from that estimate:

$$\hat{S} = \mathbb{E}\,S + \operatorname{sd}(S)\,h / w_{1\parallel}, \qquad \hat{y}_i = \left\lfloor (\hat{S} + \tfrac{1}{2}) / b^{n-i} \right\rfloor \bmod b.$$

The decoder's estimate is noisy because h also contains the second term $w_{1\perp}\,\eta$, orthogonal to the magnitude direction, whose contribution to $\hat{S}$ decays as $e^{-s_n t}$ (Section B.4). The resolution onset for output digit i occurs once $\operatorname{sd}(\hat{S} - S)$ and b^{n-i} become of the same order. This gives

$$\boxed{t_i = \frac{1}{s_n}\left[i \ln b + \ln(c\,s_n)\right] + O(e^{-2 s_n t_i})} \tag{3}$$

where c is a constant set by the initialisation. The resolution time therefore grows linearly with digit position i, so digits are resolved from most to least significant (figure 2). The intuition here is that reading digits from a single scalar encoding of the sum is easy for significant digits, but difficult for less significant ones, as an inaccurate encoding will disproportionately affect them. Computing the sum with vector addition is a shortcut that allows a model to quickly learn to compute the more significant digits, before the less significant ones.

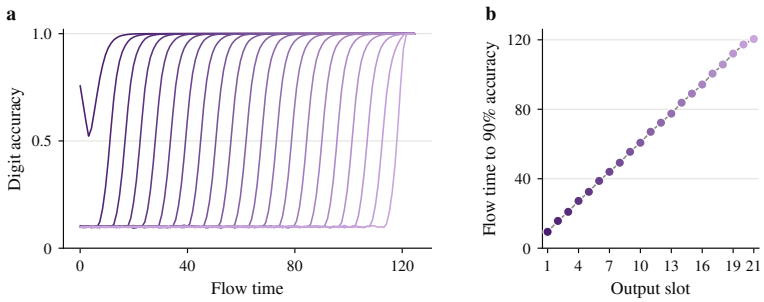


Figure 2: Digit accuracy of a perfect decoder on the representation learned in a two layer linear network trained on 20-digit addition. **(a)** Accuracy per output digit slot. **(b)** Gradient-flow time for each digit to reach 90% accuracy. The decoder becomes accurate on the more significant digits first.

4 The Transformer Exhibits Estimation-Like Properties

Reversing the digit order. The magnitude representation in the linear model from Section 3 is only learned because of the overflow digit, and can be shown to not emerge without it (Section B.3). If a similar mechanism plays a role in the reverse digit order observed in the encoder transformer, then removing the overflow digit would stop it from learning in order of most to least significant digits. In figure 3a it can be seen that without the overflow, the model is significantly halted in its ability to learn the task. The task can be made easier at the ones position, by replacing the target digit by a zero when there is no carry, and a one where there is a carry, turning it into a binary task that still retains all information necessary for computing the next digit. In this case the model learns the digits from least significant to most significant, as can be seen in figure 3b and c.

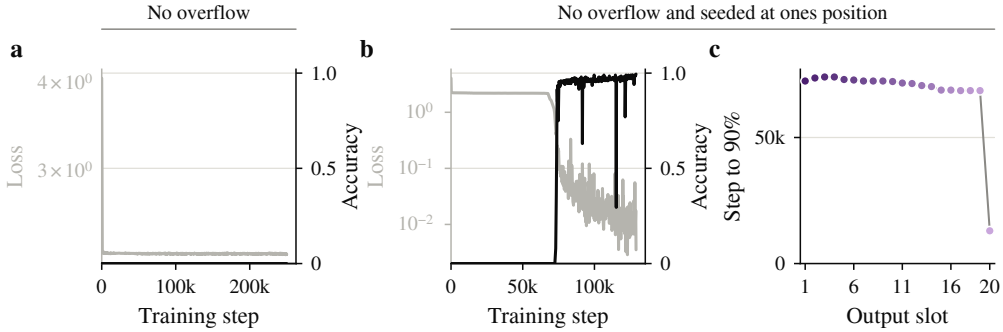


Figure 3: **Removing the overflow stops the model from learning in reverse order. (a)** Training loss and accuracy of the encoder transformer trained on a modified version of 20-digit addition, where the overflow (most significant output position) has been removed from the loss function. **(b)** Training loss and accuracy for the same setting, where additionally the least-significant output digit has been replaced by a binary value encoding its carry. **(c)** Number of steps to reach 90% accuracy at each output digit position. Note that less significant digits are learned earlier.

Attention patterns. A signature of transformers implementing the standard digit-wise algorithm, i.e. summing digits at every corresponding position and propagating a carry from the digit below, is the emergence of heads with a diagonal or position-below attention pattern. The diagonal heads look at the same position to perform the base summation, whereas position-below heads look at the position immediately prior to verify if a carry must be added (Quirke & Barez, 2024; Kruthoff, 2024; Cho et al., 2025). In figure 4, it can be seen that the model trained on standard 20-digit addition shows no evidence for them. Additionally, an imbalance can be observed, where heads at each digit mainly attend to digits with higher significance, more consistent with estimation than a bottom-up

calculation. Contrast this to the model trained on the task without an overflow digit, which learned the digits from least to most significant. Here multiple heads matching exactly the diagonal or carry-like position-below patterns were found, and the attention imbalance never went above the baseline 0.5. In general, attention patterns do not guarantee a head is performing a specific computation, however, a causal mediation analysis confirms the diagonal and position-below heads do perform the respective digit-wise computations (see Section C.2).

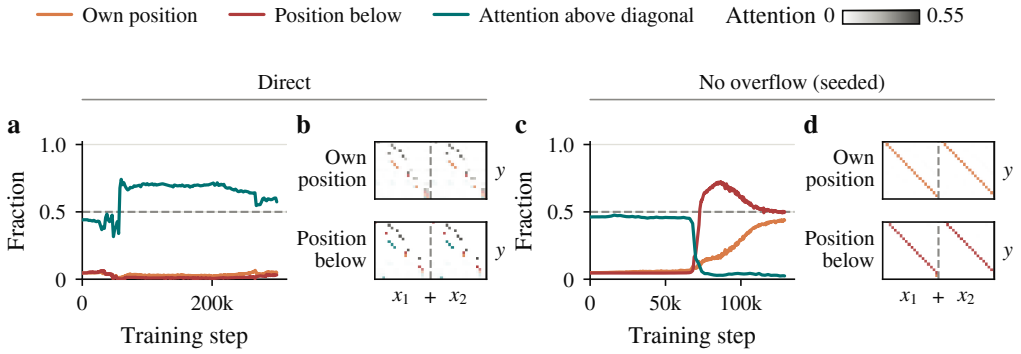


Figure 4: **Removing the overflow and supplying a carry target shifts attention towards digit-wise computation. (a)** Attention share during training on the diagonal (orange), on positions one below the diagonal (red), and above the diagonal (teal), for the final layer of the model trained on vanilla 20-digit addition. **(b)** Input-output attention of heads with the highest diagonal attention and position-below attention respectively, for the model trained on vanilla 20-digit addition. **(c)** Attention shares for the seeded no-overflow task. **(d)** Highest scoring diagonal and position-below heads for the seeded no-overflow task. Only the last two layers showed structure. Full attention plots in Section C.2.

Behavioural comparison. Both models generalise as they score well on new datapoints generated for the task (Section C.1). However, datapoints with very long carry-chains (e.g. $999999 + 000001$) are exceedingly rare, so comparing accuracy on them may still provide differences at a behavioural level. If a model implements the carry-based digit-wise algorithm, it can only propagate a carry through a fixed number of positions, bounded by the model's depth. Accuracy should therefore stay high up to that bound and then drop sharply for one position longer. Conversely, an estimation strategy predicts each digit from a heuristic whose evidence about a distant carry weakens with distance, so accuracy should decline gradually as the carry-chain gets longer. The standard addition model's behaviour is more consistent with the estimation strategy, and the no-overflow model more consistent with the digit-wise algorithm (figure 5a). Additionally, we find that only the standard addition model scores poorly on misleading problems for estimation, where there are many high digits nearby suggesting a carry, but no carry actually occurs (Section C.2).

Transient shared representation. The learned algorithm is not consistent with the standard digit-wise computation, instead sharing some properties with estimation. To understand more how this estimation-like algorithm is learned, we train linear probes to decode each digit's value from the representation at every position (figure 5b and c). Decodability is defined as the accuracy of the probe, normalised so 0 is the accuracy on the untrained model, and 1 is perfect accuracy. At convergence we see that digits are mainly represented at their own position, as expected for any accurate addition algorithm. But early in training, when the model is learning mostly just the significant digits, the overflow is encoded at *all* positions, not just its own. This shared representation is consistent with the overflow-derived magnitude mode from Section 3 used for learning the estimation strategy. Note, however, that it is only transient here, whereas for the linear model it persists.

5 Curriculum Learning Can Steer the Strategy

In Section 4 we have seen that the model trained on 20-digit addition shows properties matching an estimation strategy rather than a digit-wise algorithm. The digit learning order can be reversed, by

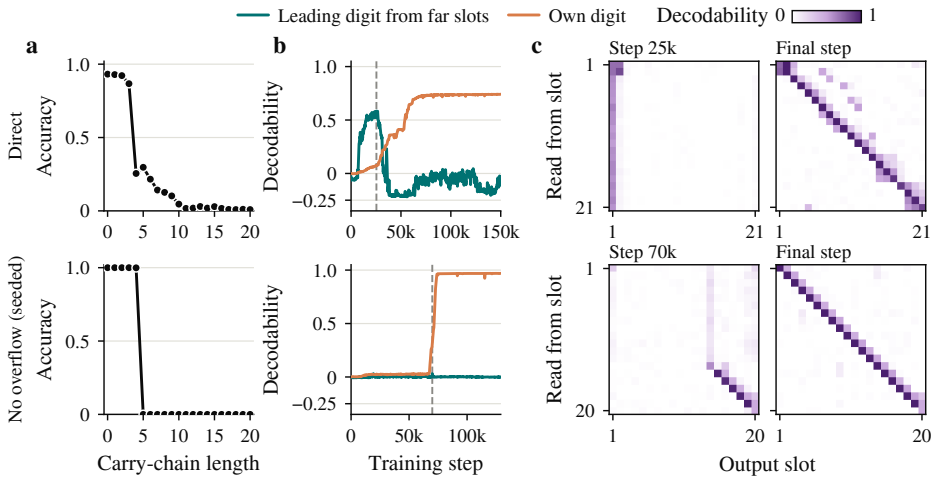


Figure 5: **Carry-chain accuracy is more estimation-like and the model has a transient shared overflow representation. (a)** Accuracy on datapoints as a function of the maximum carry-chain contained. The model trained on unmodified 20-digit addition shows exponential decay. Contrast this with the model trained on the task with the overflow removed, where the accuracy is at 100% and then instantly drops to zero at length 5. **(b)** Average decodability of each slots from its own representation (orange) and of the overflow digit at representations of slots 4 or higher (teal). **(c)** Digit-wise decodability early in training (where average digit accuracy has risen to 0.2) and at convergence. Only the model trained on standard addition shows the transient shared representation.

using a modified version of the task, shifting the learned algorithm towards a digit-wise computation. One may wonder if using a different learning approach on the *same* task can also result in a shift of strategy. Lee et al. (2024) reversed the autoregressive generation order to achieve this. This would not work for our non-autoregressive setting, and so we turn to curriculum learning instead.

Increasing length curriculum. We use a curriculum of increasing digit counts. The curriculum starts with only numbers with a single digit, then numbers with two digits, and so on, scaling up to the full 20-digit task. When the model reaches 99% accuracy on the current stage, the curriculum moves on to the next stage. Numbers are padded from the left with zeros, so digit positions stay in place as the numbers get larger. Training with the curriculum does not change the final task that must be learned, but effectively forces the model to learn the least significant digits first (figure 6a).

Consequences for the learned strategy. When the model is trained with the curriculum, similar results as for the overflow-removed task from Section 4 can be observed, showing a move towards a digit-wise algorithm (figure 6b-d and Section C.2). We also find that the model solves the task faster, especially if we make the number of digits it has to solve for longer (see Section C.3).

Generalisation performance. The standard digit-wise algorithm applies the same local rule at every position, regardless of which digit or carry value occurs there. It should therefore generalise to datapoints containing an input digit, output digit, or carry-into-position combination withheld during training. The curriculum-trained model shows higher accuracy than the directly trained model on these held-out combinations (figure 7a). Additionally, a model that has learned the standard digit-wise algorithm should be easily adaptable to tasks with a shared computational structure, whereas one that has learned a statistical shortcut should gain little from fine-tuning. Figure 7b shows that the curriculum-trained model fine-tunes significantly faster on tasks sharing addition's computational structure. It also changes its weights less during fine-tuning (figure 7c).

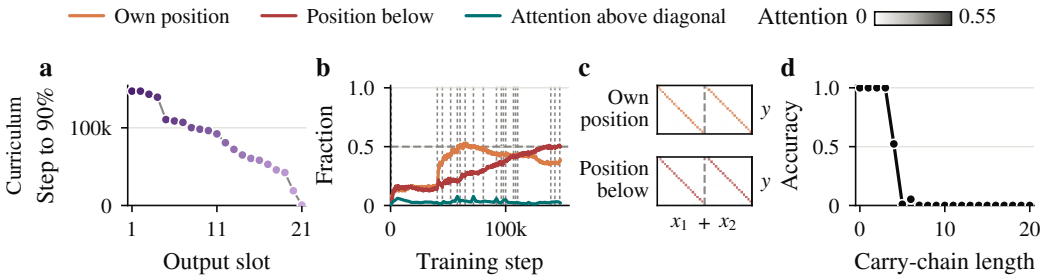


Figure 6: **Curriculum redirects acquisition towards digit-wise computation.** **(a)** Number of steps to reach 90% accuracy at each output digit position, for the model trained on a curriculum of increasing digit count. **(b)** Attention share during training on the diagonal (orange), on one position below where a carry computation would need to look (red), and above the diagonal (teal). Dashed lines are the transition points to the next curriculum stage. **(c)** Input-output attention of heads with the highest diagonal attention and position-below attention. **(d)** Accuracy on datapoints as a function of the longest carry-chain. These signatures resemble the seeded no-overflow setting rather than direct training.

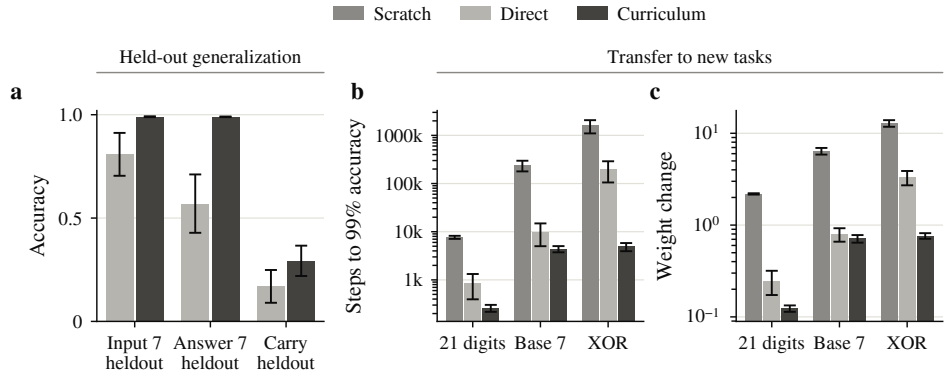


Figure 7: **Curriculum learning improves generalisation performance.** Generalisation performance between the model trained directly on 20-digit addition, versus with a curriculum from digit length 1 running up to 20. **(a)** Accuracy on held-out data during training, where the input 7 was never observed on position 3, the answer 7 was never observed on position 3, or a carry into position 3 never occurred. **(b)** Number of training steps to 99% accuracy on 21-digit addition, 20-digit addition in base 7, and 20-digit position-wise XOR, where the model is either trained from scratch, or fine-tuned from the direct or curriculum model. **(c)** L2 norm of the difference between the model's weights at initialisation versus convergence.

6 Additional Settings

Other architectures. MLPs, which like the non-autoregressive transformer have no forced digit order, also turn out to learn digits from most significant to least. Autoregressive transformers and RNNs both show high accuracy on any carry-chain, when fed the sequence from least to most significant digits. A possible explanation here is that in this generation order a model can apply carry operations n times, where n is the number of digits, thus bypassing depth-limitations (figure 22).

Large language models. Even frontier models still fail consistently on addition of long numbers (Bie et al., 2026). Figure 8 shows the accuracy of claude-haiku-4-5 on addition problems per digit, showing the model fails disproportionately on less significant digits. Prefill was used to emulate teacher-forcing, to avoid a confound where mistakes early in the sequence cause mistakes later.

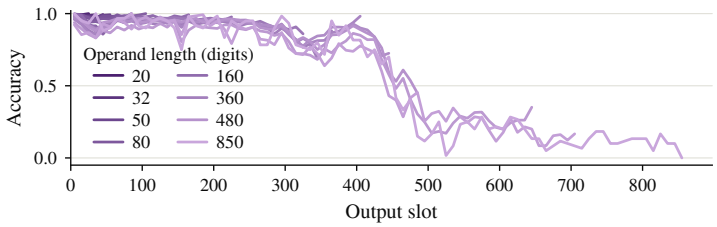


Figure 8: **Claude makes more mistakes on less significant digits.** Per digit accuracy of claude-haiku-4-5 on addition problems for different digit counts, each averaged over 200 problems.

Data. Subtraction and binary addition are also learned from most significant to least significant digits, and for both the increasing length curriculum speeds up learning (figure 23). For addition on a finite dataset, we observe a transition from overfitting to generalisation, as dataset size increases, with an intermediate regime, similar to (Power et al., 2022), where generalisation lags well behind fitting the training data. Carry and diagonal heads emerge with the transition to generalisation (figure 21).

7 Discussion

Final accuracy alone does not identify the computation learned by a model. Here we saw that a transformer trained on addition can learn a solution more akin to estimation, or one more like digit-wise computation, while being accurate on newly generated data in both cases. The exact solution of our linear network model shows how the statistics of the task favour a scalar representation of the sum's magnitude. With an ideal decoder, errors in this representation disproportionately affect the less significant digits, producing most-significant-digit-first acquisition. Our results suggest that the computation a network acquires is not fixed by the task, but can be influenced by which statistical regularities are easiest to exploit early in training.

The learned strategy can be steered. Both removal of the overflow digit and an increasing length curriculum, eliminate the signal in the data allowing for early learning of an estimation strategy. In both cases, we observed a mechanistic shift away from statistical shortcuts, and towards the more canonical digit-wise addition algorithm, although bounded by depth. Curriculum learning is particularly important here, as it changes the learned computation without modifying the architecture or final task. The resulting solution performs better on held-out patterns and adapts faster to other tasks. Curriculum learning therefore does not only speed up training, but functions as a tool of implicit algorithm steering, changing the computation acquired and improving generalisation.

This dependence on training history also connects to developmental research in arithmetic learning. Strategy-choice models and overlapping-waves theory emphasise how experience changes which strategies are used (Siegler & Shrager, 1984; Siegler, 1996). In our setting, this dependence emerges through gradient-based learning, without an explicit strategy-selection mechanism.

7.1 Limitations

Although the linear theory shows that an estimation shortcut exists on this task, it should not be taken as a literal interpretation of the algorithm the transformer learns. For example, the shared representation for the overflow is only transiently present in the transformer, whereas this remains in the linear model at convergence. Similarly, steering the model shows a shift towards the canonical digit-wise algorithm but does not construct that exact algorithm, as the model still fails on long carry-chains. Finally, using curriculum learning for implicit algorithm steering should only be taken as an example to study how a training approach affects a learned strategy, but cannot always be easily applied for improving neural network generalisation, as designing the right curriculum requires mechanistic insight into the task.

AI USE STATEMENT

Large language models were used for finding relevant literature, editing text, and implementing methods. In particular, Claude code was used as a coding assistant for writing code to perform some of the experiments. Generated code was verified and tested. We take responsibility for the final content of this work, including text, claims or artefacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All code used for experiments will be made public upon publication, including all relevant seeds for generating figures from the paper.

REFERENCES

Emmanuel Abbe, Enric Boix-Adserà, Matthew S. Brennan, Guy Bresler, and Dheeraj Nagaraj. The staircase property: How hierarchical structure can guide deep learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/e2db7186375992e729165726762cb4c1-Abstract.html.

Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep networks. In *The Seventh International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BkeStsCcKQ.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022. doi: 10.1162/coli_a_00422. URL https://aclanthology.org/2022.cl-1.7/.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 41–48. ACM, 2009. doi: 10.1145/1553374.1553380.

Raphaël Berthier, Andrea Montanari, and Kangjie Zhou. Learning time-scales in two-layers neural networks. *Foundations of Computational Mathematics*, 25(5):1627–1710, 2025. doi: 10.1007/s10208-024-09664-9. URL https://link.springer.com/article/10.1007/s10208-024-09664-9.

Alex Bie, Travis Dick, Alex Kulesza, Prabhakar Raghavan, Vinod Raman, and Sergei Vassilvitskii. Ai-rithmetic. *arXiv preprint arXiv:2602.10416*, 2026.

Rishi Bommasani et al. On the opportunities and risks of foundation models, 2021. URL https://arxiv.org/abs/2108.07258.

John Seely Brown and Richard R. Burton. Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science*, 2(2):155–192, 1978. doi: 10.1207/s15516709cog0202_4.

Hanseul Cho, Jaeyoung Cha, Pranjal Awasthi, Srinadh Bhojanapalli, Anupam Gupta, and Chulhee Yun. Position coupling: Improving length generalization of arithmetic transformers using task structure. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0701. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/27aa3a0e6d63db269977bb2df5607cb8-Abstract-Conference.html.

Hanseul Cho, Jaeyoung Cha, Srinadh Bhojanapalli, and Chulhee Yun. Arithmetic transformers can length-generalize in both operand length and count. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/c6c29e590e3c62e37e6b39cdd6baf2e8-Abstract-Conference.html.

Bilal Chughtai, Lawrence Chan, and Neel Nanda. A toy model of universality: Reverse engineering how networks learn group operations. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 6243–6267. PMLR, 2023. URL https://proceedings.mlr.press/v202/chughtai23a.html.

Elisabetta Cornacchia and Elchanan Mossel. A mathematical model for curriculum learning for parities. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 6402–6423. PMLR, 2023. URL https://proceedings.mlr.press/v202/cornacchia23a.html.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*, 23(226):1–61, 2022. URL https://jmlr.org/papers/v23/20-1335.html.

Dana Ganor-Stern. Do exact calculation and computation estimation reflect the same skills? developmental and individual differences perspectives. *Frontiers in Psychology*, 9:1316, 2018.

Atticus Geiger, Duligur Ibeling, Amir Zur, Maheep Chaudhary, Sonakshi Chauhan, Jing Huang, Aryaman Arora, Zhengxuan Wu, Noah Goodman, Christopher Potts, and Thomas Icard. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 26(83):1–64, 2025. URL https://jmlr.org/papers/v26/23-0058.html.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020. doi: 10.1038/s42256-020-00257-z.

Camilla K. Gilmore, Shannon E. McCarthy, and Elizabeth S. Spelke. Symbolic arithmetic knowledge without instruction. *Nature*, 447:589–591, 2007. doi: 10.1038/nature05850. URL https://www.nature.com/articles/nature05850.

Elise Klein, Klaus Willmes, Katharina Dressel, Frank Domahs, Guilherme Wood, Hans-Christoph Nuerk, and Korbinian Moeller. Categorical and continuous-disentangling the neural correlates of the carry effect in multi-digit addition. *Behavioral and Brain Functions*, 6(1):70, 2010.

Jorrit Kruthoff. Carrying over algorithm in transformers, 2024. URL https://arxiv.org/abs/2401.07993.

Nayoung Lee, Kartik Sreenivasan, Jason D. Lee, Kangwook Lee, and Dimitris Papailiopoulos. Teaching arithmetic to small transformers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=dsUB4bst9S.

Patrick Lemaire and Robert S. Siegler. Four aspects of strategic change: Contributions to children's learning of multiplication. *Journal of Experimental Psychology: General*, 124(1):83–97, 1995. doi: 10.1037/0096-3445.124.1.83.

Patrick Lemaire, Mireille Lecacheur, and Fernand Farioli. Children's strategy use in computational estimation. *Canadian Journal of Experimental Psychology*, 54(2):141–148, 2000. doi: 10.1037/h0087336. URL https://pubmed.ncbi.nlm.nih.gov/10881397/.

David Lindner, János Kramár, Sebastian Farquhar, Matthew Rahtz, Thomas McGrath, and Vladimir Mikulik. Tracr: Compiled transformers as a laboratory for interpretability. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-1649. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/771155abaae744e08576f1f3b4b7ac0d-Abstract-Conference.html.

Sean McLeish, Arpit Bansal, Alex Stein, Neel Jain, John Kirchenbauer, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, Jonas Geiping, Avi Schwarzschild, and Tom Goldstein. Transformers can do arithmetic with the right embeddings. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3430. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/c35986bc1ee29b31c1011481b77fe540-Abstract-Conference.html.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=9XFSbDPmdW.

Yaniv Nikankin, Anja Reusch, Aaron Mueller, and Yonatan Belinkov. Arithmetic without algorithms: Language models solve math with a bag of heuristics. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/8c5f30296296d2ae402ebbd09aaa9c12-Abstract-Conference.html.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Philip Quirke and Fazl Barez. Understanding addition in transformers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/f115f619b62833aadc5acb058975b0e6-Abstract-Conference.html.

Philip Quirke, Clement Neo, and Fazl Barez. Understanding addition and subtraction in transformers, 2024. URL https://arxiv.org/abs/2402.02619.

Luca Saglietti, Stefano Sarao Mannelli, and Andrew Saxe. An analytical theory of curriculum learning in teacher-student networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 21113–21127, 2022. doi: 10.52202/068431-1535. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/84bad835faaf48f24d990072bb5b80ee-Abstract-Conference.html.

Stefano Sarao Mannelli, Yaraslau Ivashynka, Andrew M. Saxe, and Luca Saglietti. Tilting the odds at the lottery: the interplay of overparameterisation and curricula in neural networks. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 34586–34602. PMLR, 2024. URL https://proceedings.mlr.press/v235/mannelli24a.html.

Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013. doi: 10.48550/arXiv.1312.6120. URL https://arxiv.org/abs/1312.6120.

Jeff Shrager and Robert S. Siegler. SCADS: A model of children's strategy choices and strategy discoveries. *Psychological Science*, 9(5):405–410, 1998. doi: 10.1111/1467-9280.00076.

Robert S. Siegler. The perils of averaging data over strategies: An example from children's addition. *Journal of Experimental Psychology: General*, 116(3):250–264, 1987. doi: 10.1037/0096-3445.116.3.250.

Robert S. Siegler. *Emerging Minds: The Process of Change in Children's Thinking*. Oxford University Press, New York, 1996. doi: 10.1093/oso/9780195077872.001.0001.

Robert S. Siegler and Jeffrey Shrager. Strategy choices in addition and subtraction: How do children know what to do? In Catherine Sophian (ed.), *Origins of Cognitive Skills*, pp. 229–293. Lawrence Erlbaum Associates, Hillsdale, NJ, 1984. URL https://siegler.tc.columbia.edu/wp-content/uploads/2019/11/1984-Siegler-Shrager.pdf.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5998–6008. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 12388–12401. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92650b2e92217715fe312e6fa7b90d82-Paper.pdf.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: A circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=NpsVSN6o4ul.

Daphna Weinshall, Gad Cohen, and Dan Amir. Curriculum learning by transfer learning: Theory and experiments with deep networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5238–5246. PMLR, 2018. URL https://proceedings.mlr.press/v80/weinshall18a.html.

Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? In *The Ninth International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=tW4QEInpni.

Xingyu Zhao, Darsh Sharma, Rheeya Uppaal, and Yiqiao Zhong. Shattered compositionality: Counterintuitive learning dynamics of transformers for arithmetic, 2026. URL https://arxiv.org/abs/2601.22510.

Ziqian Zhong, Ziming Liu, Max Tegmark, and Jacob Andreas. The clock and the pizza: Two stories in mechanistic explanation of neural networks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 27223–27250, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/56cbfbf49937a0873d451343ddc8c57d-Abstract-Conference.html.

Tianyi Zhou, Deqing Fu, Vatsal Sharan, and Robin Jia. Pre-trained large language models use fourier features to compute addition. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0792. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/2cc8dc30e52798b27d37b795cc153310-Abstract-Conference.html.

A Experimental Details

Task and data generation. Unless stated otherwise, addition problems are generated online, i.e. in every optimiser step a new batch is generated. For numbers with many digits this means that no training examples are seen twice and the model's performance does not reflect memorisation of a fixed dataset. For an n-digit task or curriculum stage, each of the two operands is drawn independently and uniformly from $\{1, \ldots, 10^n - 1\}$ and always left-padded with zeros to the maximum possible digit length (e.g. for curriculum stages this is based on the final tasks length, but also for direct learning the output will be padded with a zero if no carry occurs there to keep it always length $n + 1$). All operand digits and the + separator are encoded categorically using one hot vectors, so no explicit numerical-value based features are present in their representation. The target output is the sum's $n + 1$ digits written most-significant first unless stated otherwise, with the leading digit $y_0 \in \{0, 1\}$, also referred to as the overflow.

Two task edits are used to test for the role of the overflow digit and the acquisition order. For the 'No overflow' edit, y_0 is removed from the target, so the loss used during training and every accuracy metric reported are computed over $y_1, \ldots, y_n$ only. The edit 'No overflow (seeded)' additionally replaces the least-significant target digit y_n with the binary carry-value coming out of the ones position, i.e. $\lfloor (a_n + b_n)/10 \rfloor \in \{0, 1\}$. This is learnable from the ones position inputs alone, and a binary value making it relatively easy to learn, but it still contains all necessary information for computing the second least significant output digit. Subtraction, addition in other bases, and position-wise XOR all have an analogous encoding and training pipeline.

Architecture and optimisation. The model used is an encoder transformer unless stated otherwise. It consists of four self-attention layers, four heads per layer, embedding dimension 256, feedforward dimension 2048, and sinusoidal positional encoding. The input sequence consisting of both operand's digits and the + symbol, is embedded and then concatenated with $n+1$ learned answer-slot query vectors, one per output digit. The entire sequence passes through the same self-attention stack without causal masking, so every position can freely attend to every other in a single parallel forward pass, and no output digit is conditioned on any other. Parameter optimisation is performed using AdamW with learning rate 5×10^{-5} and weight decay 0.01, on the mean cross-entropy loss over all output digit slots, with batch size 64. Architectures and hyperparameters for the MLP, RNN and autoregressive-decoder are in Section C.4.

Curriculum schedule. The increasing-length curriculum trains the model in stages of increasing input lengths $n = 1, \ldots, 20$ in order. Operands are always left-padded with zeros to have the final 20-digit width, and so a digit's position in the input and output sequence is fixed across the stages. Every 200 training steps we evaluate the model on a batch of 200 examples drawn from the training distribution but held fixed for that stage. If the exact-match accuracy on the held-out evaluation batch exceeds 99%, the curriculum moves on to the next stage. The threshold was fixed before any curriculum experiments and never tuned.

Large language model evaluation. The Claude results from figure 8 query `claude-haiku-4-5` via prefill in the raw Messages API, without system prompt, tools, and extended thinking all turned off, so each answer is produced in a single parallel pass without any harness. Temperature is set to 0. The prompt used is

```
"Reply with only the sum as digits, nothing else.\n\n{a} + {b} ="
```
(4)

Both the operands always have the same digit count, and digits are drawn uniformly. Results are averaged over 200 problems for each digit length.

Finite dataset. For the finite dataset results in Section C.4, a pool of N fixed examples is generated at the start of training and accuracy is tracked both on this pool and another disjoint held-out pool of the same size.

B Exact Learning Dynamics of The Linear Network

B.1 Correlation matrices

Setup. Assume we are given input numbers a, a' in base $b \geq 2$ with number of digits $n \geq 1$, and entries sampled i.i.d. uniform on $\{0, \ldots, b-1\}$. To simplify computations, we let $z \in \{0, \ldots, b-1\}^{2n}$ be the concatenation of the digits of a and a', each numbered from the most significant, and define the place-value exponent

$$\iota(p) = \begin{cases} n-p & \text{if } 1 \leq p \leq n, \\ 2n-p & \text{if } n < p \leq 2n, \end{cases} \tag{5}$$

so that z_p has place value $b^{\iota(p)}$, and ι runs over $\{n-1, \ldots, 0\}$ once for the digits of a and once for the digits of a' as p ranges over $\{1, \ldots, 2n\}$. With S, y_i as in equation 1 and $y_{\text{ovf}} := y_0$, write $R := S \bmod b^n$, so that $S = b^n y_{\text{ovf}} + R$ and $y_i = \lfloor R/b^{n-i} \rfloor \bmod b$ for $i \geq 1$.

The inputs $x \in \mathbb{R}^D$ and targets $y \in \mathbb{R}^K$ with dimensions $D = 2nb$ and $K = 1 + nb$ respectively, are encoded as centered one-hot encoded vectors: $x_{p,k} = \sqrt{b}\,(\mathbf{1}\{z_p = k\} - \frac{1}{b})$, $[y]_{\text{ovf}} = y_{\text{ovf}} - \mathbb{E}\, y_{\text{ovf}}$ and $[y]_{i,k} = \mathbf{1}\{y_i = k\} - \frac{1}{b}$ for $i \geq 1$. The overflow basis vector is written as e_{ovf}, which is the output vector that is zero everywhere except one at the overflow location. We write the correlation matrices as

$$\Sigma_{xx} = \mathbb{E}[xx^\top], \qquad \Sigma_{yx} = \mathbb{E}[yx^\top] = \Sigma_{xy}^\top, \qquad \Sigma_{yy} = \mathbb{E}[yy^\top]. \tag{6}$$

The network is given by $\hat{y} = W_2 W_1 x$, $W_1 \in \mathbb{R}^{1 \times D}$, $W_2 \in \mathbb{R}^{K \times 1}$, $h(x) = W_1 x$, and learning dynamics are gradient flow on $L = \frac{1}{2}\,\mathbb{E}\,\|\hat{y} - y\|^2$:

$$\dot{W}_1 = W_2^\top (\Sigma_{yx} - W_2 W_1 \Sigma_{xx}), \qquad \dot{W}_2 = (\Sigma_{yx} - W_2 W_1 \Sigma_{xx}) W_1^\top, \tag{7}$$

where we absorbed the learning rate into the time variable.

Output statistics. For fixed a, the map $a' \mapsto (a + a') \bmod b^n$ is a bijection of $\mathbb{Z}_{b^n}$, so R is uniform given fixed a, and hence uniform and independent of a, and using a similar argument also to a'. Since the map assigning numbers to their digit sequences $\mathbb{Z}_{b^n} \to \{0, \dots, b-1\}^n$ is also a bijection, the non-overflow output digits $(y_1, \dots, y_n)$ are consequently i.i.d. uniform and independent of every z_p.

Using the fact that $r + 1$ pairs satisfy $a + a' = r$, we can compute the statistics

$$\mathbb{E}\, y_{\text{ovf}} = 1 - \sum_{r=0}^{b^n-1} \frac{r+1}{b^{2n}} = \frac{1 - b^{-n}}{2}, \qquad \operatorname{Var} y_{\text{ovf}} = \frac{\kappa_n}{4}, \qquad \operatorname{tr} \Sigma_{yy} = \frac{\kappa_n}{4} + \frac{n(b-1)}{b}, \tag{8}$$

where $\kappa_n := 1 - b^{-2n}$ and the last equality also uses $\operatorname{Var} \mathbf{1}\{y_i = k\} = \frac{1}{b}(1 - \frac{1}{b})$ for each of the nb digit coordinates.

Input correlation. Since the input digits are independent and uniform,

$$\big(\Sigma_{xx}\big)_{(p,k),(q,l)} = b \operatorname{Cov}\big(\mathbf{1}\{z_p = k\}, \mathbf{1}\{z_q = l\}\big) = b\, \delta_{pq}\Big(\tfrac{1}{b}\delta_{kl} - \tfrac{1}{b^2}\Big) = \delta_{pq}\Big(\delta_{kl} - \tfrac{1}{b}\Big), \tag{9}$$

so that

$$\Sigma_{xx} = \Pi := \bigoplus_{p=1}^{2n} \Big(I_b - \tfrac{1}{b}\mathbf{1}_b\mathbf{1}_b^\top\Big). \tag{10}$$

which is the orthogonal projection onto the vectors whose sum over the embedding dimension is zero for every position. If we write $x_p \in \mathbb{R}^b$ for the block of x belonging to digit p, since the inputs are centered we have $\mathbf{1}_b^\top x_p = \sum_k x_{p,k} = 0$, and so $x \in \operatorname{range} \Pi$.

Input–output correlation. All non-overflow digit rows of Σ_{yx} are zero because of independence:

$$\mathbb{E}\big[[y]_{i,l}\, x_{p,k}\big] = \sqrt{b} \operatorname{Cov}\big(\mathbf{1}\{y_i = l\}, \mathbf{1}\{z_p = k\}\big) = 0. \tag{11}$$

For the overflow, we write $y_{\text{ovf}} = (S - R)/b^n$ and $S = \sum_q b^{\iota(q)} z_q$. Using the independence of z_p with R and any z_q with $q \neq p$, the overflow row from Σ_{yx} can be computed as:

$$\begin{aligned}
\mathbb{E}\big[[y]_{\text{ovf}}\, x_{p,k}\big] &= \mathbb{E}\left[\left(\frac{S-R}{b^n} - \mathbb{E}\, y_{\text{ovf}}\right)\sqrt{b}\,\big(\mathbf{1}\{z_p = k\} - \tfrac{1}{b}\big)\right] \\
&= \frac{\sqrt{b}}{b^n}\sum_q b^{\iota(q)}\, \mathbb{E}\left[z_q\,\big(\mathbf{1}\{z_p = k\} - \tfrac{1}{b}\big)\right] \\
&= \frac{\sqrt{b}\, b^{\iota(p)}}{b^n}\left(k \cdot \mathbb{E}\big[\mathbf{1}\{z_p = k\}\big] - \frac{1}{b}\,\mathbb{E}[z_p]\right) \\
&= b^{1/2-n+\iota(p)}\left(\frac{k}{b} - \frac{b-1}{2b}\right) \\
&= b^{-1/2-n}\gamma_{p,k}
\end{aligned}$$

where $\gamma_{p,k} := b^{\iota(p)}(k - \frac{b-1}{2})$ with norm:

$$\begin{aligned}
\|\gamma\|^2 &= \Big(\sum_p b^{2\iota(p)}\Big)\Big(\sum_k \big(k - \tfrac{b-1}{2}\big)^2\Big) \\
&= \Big(2\big(1 + b^2 + \cdots + b^{2(n-1)}\big)\Big)\Big(b \cdot \frac{b^2-1}{12}\Big) \\
&= \frac{2\, b^{2n}\kappa_n}{b^2-1} \cdot \frac{b(b^2-1)}{12} \\
&= \frac{b^{2n+1}\kappa_n}{6}.
\end{aligned} \tag{12}$$

Thus Σ_{yx} is rank one:

$$\Sigma_{yx} = s_n \, e_{\text{ovf}} v^\top, \qquad v = \frac{\gamma}{\|\gamma\|}, \qquad s_n = b^{-1/2-n}\|\gamma\| = \sqrt{\frac{\kappa_n}{6}}. \tag{13}$$

With a similar argument as before with x, it follows from $\sum_k \gamma_{p,k} = 0$ that $v \in \operatorname{range} \Pi$.

Projecting input on the magnitude direction returns the sum. Using $\sum_k \gamma_{p,k} = 0$, we compute

$$\sum_k \gamma_{p,k} \, x_{p,k} = \sqrt{b}\, b^{\iota(p)} \sum_k \Big(k - \frac{b-1}{2}\Big)\Big(\mathbf{1}\{z_p = k\} - \frac{1}{b}\Big) = \sqrt{b}\, b^{\iota(p)}\Big(z_p - \frac{b-1}{2}\Big), \tag{14}$$

giving

$$\begin{aligned}
\gamma^\top x &= \sqrt{b} \sum_p b^{\iota(p)} \Big(z_p - \frac{b-1}{2}\Big) && (15)\\
&= \sqrt{b}\Big(\sum_p b^{\iota(p)} z_p - \sum_p b^{\iota(p)} \tfrac{b-1}{2}\Big) && (16)\\
&= \sqrt{b}\big(S - (b^n - 1)\big) && (17)\\
&= \sqrt{b}\,(S - \mathbb{E}\, S), && (18)
\end{aligned}$$

where we use $\sum_p b^{\iota(p)} z_p = S$ and $\sum_p b^{\iota(p)} \frac{b-1}{2} = b^n - 1 = \mathbb{E}\, S$. From this it follows that

$$v^\top x = \frac{\gamma^\top x}{\|\gamma\|} = \frac{\sqrt{b}\,(S - \mathbb{E}\, S)}{\|\gamma\|} = \frac{\sqrt{b}\,(S - \mathbb{E}\, S)}{\sqrt{b}\, b^n s_n} = \frac{S - \mathbb{E}\, S}{\operatorname{sd}(S)}, \tag{19}$$

where we use $\operatorname{Var} S = 2 \operatorname{Var} a = (b^{2n} - 1)/6$ and $\|\gamma\|/\sqrt{b} = b^n s_n = \operatorname{sd}(S)$. So the inner product between the magnitude direction and input is the standardised sum.

B.2 Solution of the learning dynamics

Training loss. For any $W_1 \in \mathbb{R}^{1\times D}$, $W_2 \in \mathbb{R}^{K\times 1}$, using $\Pi^2 = \Pi$ and $\Sigma_{yx}\Pi = \Sigma_{yx}$, the loss can be decomposed as

$$\begin{aligned}
\tfrac{1}{2}\,\mathbb{E}\,\|W_2 W_1 x - y\|^2 &= \tfrac{1}{2}\big[\operatorname{tr}(W_2 W_1 \Sigma_{xx} W_1^\top W_2^\top) - 2\operatorname{tr}(W_2 W_1 \Sigma_{xy}) + \operatorname{tr}\Sigma_{yy}\big]\\
&= \tfrac{1}{2}\big[\operatorname{tr}(W_2 W_1 \Pi W_1^\top W_2^\top) - 2\operatorname{tr}(W_2 W_1 \Sigma_{xy}) + \operatorname{tr}\Sigma_{yy}\big]\\
&= \tfrac{1}{2}\Big[\operatorname{tr}(W_2 W_1 \Pi W_1^\top W_2^\top) - 2\operatorname{tr}(W_2 W_1 \Pi \Sigma_{xy}) + \operatorname{tr}(\Sigma_{yx}\Sigma_{xy})\Big] + \tfrac{1}{2}\big(\operatorname{tr}\Sigma_{yy} - s_n^2\big)\\
&= \tfrac{1}{2}\big\|W_2 W_1 \Pi - \Sigma_{yx}\big\|^2 + L_\infty,\\
L_\infty &:= \tfrac{1}{2}\big(\operatorname{tr}\Sigma_{yy} - s_n^2\big) = \tfrac{1}{2}\Big(\frac{\kappa_n}{4} + \frac{n(b-1)}{b} - \frac{\kappa_n}{6}\Big) = \frac{1}{2}\Big(\frac{\kappa_n}{12} + \frac{n(b-1)}{b}\Big).
\end{aligned} \tag{20}$$

So $L \geq L_\infty > 0$ for any possible W_1, W_2, and therefore the loss will never go to zero during training.

Rank one dynamics. With $\Sigma_{xx} = \Pi$, the dynamics equation 7 become

$$\dot{W}_1 = W_2^\top \Sigma_{yx} - W_2^\top W_2 W_1 \Pi, \qquad \dot{W}_2 = \Sigma_{yx} W_1^\top - W_2 W_1 \Pi W_1^\top. \tag{21}$$

Right-multiplying the first equation by $(I - \Pi)$ shows

$$\dot{W}_1 (I - \Pi) = W_2^\top \underbrace{\Sigma_{yx}(I - \Pi)}_{=0} - W_2^\top W_2 W_1 \underbrace{\Pi(I - \Pi)}_{=0} = 0, \tag{22}$$

so the component of W_1 in $\ker \Pi$ never changes. From $x \in \operatorname{range}\Pi$ we get $W_1 x = W_1 \Pi x$, meaning that the $\ker \Pi$ component of W_1 is irrelevant to the dynamics. Without loss of generality, we take the initial condition $W_1^0(I - \Pi) = 0$, so that $W_1 \Pi = W_1$ holds for all t (for a general W_1^0, replace W_1 by $W_1\Pi$). This lets us simplify

$$W_2^\top W_2 W_1 \Pi = W_2^\top W_2 (W_1 \Pi) = \|W_2\|^2 W_1, \qquad W_2 W_1 \Pi W_1^\top = W_2 (W_1 \Pi) W_1^\top = \|W_1\|^2 W_2, \tag{23}$$

giving

$$\dot{W}_1 = W_2^\top \Sigma_{yx} - \|W_2\|^2 W_1, \qquad \dot{W}_2 = \Sigma_{yx} W_1^\top - \|W_1\|^2 W_2. \tag{24}$$

Finally, we substitute the rank-one correlation $\Sigma_{yx} = s_n\, e_{\mathrm{ovf}} v^\top$ from equation 13 and define $w_{1\|} := W_1 v$, $w_{2\|} := e_{\mathrm{ovf}}^\top W_2$ for the components of W_1, W_2 that are aligned with the magnitude direction, resulting in the dynamics:

$$\dot{W}_1 = -\|W_2\|^2 W_1 + s_n\, w_{2\|} v^\top, \qquad \dot{W}_2 = -\|W_1\|^2 W_2 + s_n\, w_{1\|} e_{\mathrm{ovf}}. \tag{25}$$

Balanced weights. From equation 25 we can see that $\dot{W}_1 \in \mathrm{span}\{W_1, v^\top\}$ and $\dot{W}_2 \in \mathrm{span}\{W_2, e_{\mathrm{ovf}}\}$, so by uniqueness $W_1(t) \in \mathrm{span}\{W_1^0, v^\top\}$ and $W_2(t) \in \mathrm{span}\{W_2^0, e_{\mathrm{ovf}}\}$ for all t. Let $\hat{e} \perp v$ and $\hat{f} \perp e_{\mathrm{ovf}}$ be unit vectors completing these two spans (note that $\hat{e}$ must lie in $\mathrm{range}\,\Pi$), and define $w_{1\perp} := W_1 \hat{e}$, $w_{2\perp} := \hat{f}^\top W_2$ so that

$$W_1 = w_{1\|}\, v^\top + w_{1\perp}\, \hat{e}^\top, \qquad W_2 = w_{2\|}\, e_{\mathrm{ovf}} + w_{2\perp}\, \hat{f}. \tag{26}$$

From equation 25,

$$\begin{aligned} \tfrac{d}{dt}\|W_2\|^2 &= 2\langle W_2, \dot{W}_2 \rangle \\ &= 2\Big\langle W_2,\, -\|W_1\|^2 W_2 + s_n\, w_{1\|} e_{\mathrm{ovf}} \Big\rangle \\ &= -2\|W_1\|^2 \|W_2\|^2 + 2 s_n\, w_{1\|} w_{2\|} \\ &= \tfrac{d}{dt}\|W_1\|^2, \end{aligned} \tag{27}$$

so $\|W_2\|^2 - \|W_1\|^2$ is a conserved quantity. If we assume balanced weights, i.e. this is zero at initialisation, then this gives $\|W_1\| = \|W_2\| = \rho$, or equivalently $\rho^2 = w_{1\|}^2 + w_{1\perp}^2 = w_{2\|}^2 + w_{2\perp}^2$.

Projecting equation 25 onto $v, \hat{e}, e_{\mathrm{ovf}}, \hat{f}$ reduces the dynamics to four scalar equations,

$$\begin{aligned} \dot{w}_{1\|} &= \dot{W}_1 v = -\|W_2\|^2 W_1 v + s_n\, w_{2\|}(v^\top v) = -\rho^2 w_{1\|} + s_n\, w_{2\|}, \\ \dot{w}_{2\|} &= e_{\mathrm{ovf}}^\top \dot{W}_2 = -\|W_1\|^2 e_{\mathrm{ovf}}^\top W_2 + s_n\, w_{1\|}(e_{\mathrm{ovf}}^\top e_{\mathrm{ovf}}) = -\rho^2 w_{2\|} + s_n\, w_{1\|}, \\ \dot{w}_{1\perp} &= \dot{W}_1 \hat{e} = -\|W_2\|^2 W_1 \hat{e} + s_n\, w_{2\|}(v^\top \hat{e}) = -\rho^2 w_{1\perp}, \\ \dot{w}_{2\perp} &= \hat{f}^\top \dot{W}_2 = -\|W_1\|^2 \hat{f}^\top W_2 + s_n\, w_{1\|}(\hat{f}^\top e_{\mathrm{ovf}}) = -\rho^2 w_{2\perp}. \end{aligned} \tag{28}$$

Integration. We can decouple the first pair of equations in equation 28 by defining $\pi_\pm := w_{1\|} \pm w_{2\|}$, to get:

$$\begin{aligned} \dot{\pi}_+ &= \dot{w}_{1\|} + \dot{w}_{2\|} = -\rho^2(w_{1\|} + w_{2\|}) + s_n(w_{2\|} + w_{1\|}) = (s_n - \rho^2)\pi_+ \\ \dot{\pi}_- &= \dot{w}_{1\|} - \dot{w}_{2\|} = -\rho^2(w_{1\|} - w_{2\|}) + s_n(w_{2\|} - w_{1\|}) = -(s_n + \rho^2)\pi_-. \end{aligned} \tag{29}$$

with solutions

$$\begin{aligned} \pi_+ &= \pi_+^0 \exp\Big(\int_0^t \big(s_n - \rho^2(t')\big)\, dt'\Big) = \pi_+^0 e^{\,s_n t} U, \\ \pi_- &= \pi_-^0 \exp\Big(-\int_0^t \big(s_n + \rho^2(t')\big)\, dt'\Big) = \pi_-^0 e^{-s_n t} U, \end{aligned} \tag{30}$$

where $\pi_+^0 := w_{1\|}^0 + w_{2\|}^0$, $\pi_-^0 := w_{1\|}^0 - w_{2\|}^0$, and $U(t) := \exp\big(-\int_0^t \rho^2(t')\, dt'\big)$. Hence in the original system we get:

$$\begin{aligned} w_{1\|} &= \tfrac{1}{2}(\pi_+ + \pi_-) = \tfrac{1}{2}(\pi_+^0 e^{\,s_n t} + \pi_-^0 e^{-s_n t})U, \\ w_{2\|} &= \tfrac{1}{2}(\pi_+ - \pi_-) = \tfrac{1}{2}(\pi_+^0 e^{\,s_n t} - \pi_-^0 e^{-s_n t})U, \\ w_{1\perp} &= w_{1\perp}^0 U, \\ w_{2\perp} &= w_{2\perp}^0 U. \end{aligned} \tag{31}$$

Next, we write

$$\begin{aligned} A(t) := w_{1\|} w_{2\|} &= \tfrac{1}{4}\big(\pi_+^0 e^{\,s_n t} + \pi_-^0 e^{-s_n t}\big)\big(\pi_+^0 e^{\,s_n t} - \pi_-^0 e^{-s_n t}\big)U^2 = \frac{(\pi_+^0)^2 e^{2 s_n t} - (\pi_-^0)^2 e^{-2 s_n t}}{4}\, U^2, \\ \rho^2 = w_{1\|}^2 + w_{1\perp}^2 &= \Big[\tfrac{1}{4}\big(\pi_+^0 e^{\,s_n t} + \pi_-^0 e^{-s_n t}\big)^2 + (w_{1\perp}^0)^2\Big] U^2. \end{aligned} \tag{32}$$

which allows us to write a differential equation for U

$$\begin{aligned}\frac{d}{dt}U^{-2} &= -2U^{-3}\dot{U} \\ &= 2\rho^2 U^{-2} \\ &= 2\Big[\tfrac{1}{4}\big(\pi_+^0 e^{\,s_n t} + \pi_-^0 e^{-s_n t}\big)^2 + (w_{1\perp}^0)^2\Big] \\ &= \tfrac{1}{2}(\pi_+^0)^2 e^{2s_n t} + \pi_+^0\pi_-^0 + \tfrac{1}{2}(\pi_-^0)^2 e^{-2s_n t} + 2(w_{1\perp}^0)^2,\end{aligned} \tag{33}$$

which we can integrate to get

$$\begin{aligned}U(t)^{-2} &= 1 + \int_0^t \Big[\tfrac{1}{2}(\pi_+^0)^2 e^{2s_n t'} + \pi_+^0\pi_-^0 + \tfrac{1}{2}(\pi_-^0)^2 e^{-2s_n t'} + 2(w_{1\perp}^0)^2\Big] dt' \\ &= 1 + \frac{(\pi_+^0)^2(e^{2s_n t}-1) + (\pi_-^0)^2(1-e^{-2s_n t})}{4s_n} + \big(\pi_+^0\pi_-^0 + 2(w_{1\perp}^0)^2\big)\, t.\end{aligned} \tag{34}$$

Applying $W_2W_1\Pi = W_2W_1$, $\|W_2W_1\|^2 = \rho^4$, and $\langle W_2W_1, \Sigma_{yx}\rangle = s_n A$ to equation 20, we get the loss along the learning trajectory

$$\begin{aligned}L(t) &= \tfrac{1}{2}\big\|W_2W_1 - \Sigma_{yx}\big\|^2 + L_\infty \\ &= \tfrac{1}{2}\big(\|W_2W_1\|^2 - 2\langle W_2W_1, \Sigma_{yx}\rangle + \|\Sigma_{yx}\|^2\big) + L_\infty \\ &= L_\infty + \tfrac{1}{2}\big(\rho^4 - 2s_n A + s_n^2\big).\end{aligned} \tag{35}$$

Infinite time limit. For $\pi_+^0 \neq 0$, we have $U^{-2} \sim (\pi_+^0)^2 e^{2s_n t}/(4s_n)$ as t goes to infinity, and applying this to equation 31, and equation 32, we get the long-time limit:

$$\begin{aligned}U &\to \frac{2\sqrt{s_n}}{|\pi_+^0|}\, e^{-s_n t}, \\ w_{1\|} &\to \sqrt{s_n}\, \operatorname{sgn}\pi_+^0, \\ w_{2\|} &\to \sqrt{s_n}\, \operatorname{sgn}\pi_+^0, \\ w_{1\perp} &= w_{1\perp}^0 U = O(e^{-s_n t}), \\ w_{2\perp} &= w_{2\perp}^0 U = O(e^{-s_n t}), \\ \rho^2 &= w_{1\|}^2 + w_{1\perp}^2 \to s_n, \\ A &= w_{1\|}w_{2\|} \to s_n, \\ W_2W_1 &\to \Sigma_{yx}, \\ L &\to L_\infty.\end{aligned} \tag{36}$$

In the case that $\pi_+^0 = 0$ then $\pi_+ \equiv 0$ by equation 31, so $w_{1\|} = -w_{2\|}$ for all t, and hence $A = w_{1\|}w_{2\|} = -w_{2\|}^2 \leq 0$. Substituting $\pi_+^0 = 0$ into equation 32 and equation 34 gives

$$\rho^2 = \Big[\tfrac{1}{4}(\pi_-^0)^2 e^{-2s_n t} + (w_{1\perp}^0)^2\Big] U^2 = \frac{\frac{1}{4}(\pi_-^0)^2 e^{-2s_n t} + (w_{1\perp}^0)^2}{1 + (\pi_-^0)^2\big(1-e^{-2s_n t}\big)/(4s_n) + 2(w_{1\perp}^0)^2\, t} \to 0, \tag{37}$$

which decays as $1/(2t)$ if $w_{1\perp}^0 \neq 0$ and exponentially otherwise. So in this special case the model collapses. However, this lower-dimensional hyperplane is unstable since by equation 31, $\pi_+ = \pi_+^0 e^{\,s_n t} U$ grows for any $\pi_+^0 \neq 0$, so the collapse is very unlikely to occur from random initial weights.

B.3 Consequences

Hidden unit. Combining equation 19 with equation 26, we can decompose the hidden unit, for all t, as

$$h(x) = W_1 x = w_{1\|}(t)\, \frac{S - \mathbb{E}\, S}{\operatorname{sd}(S)} + w_{1\perp}(t)\, \hat{e}^\top x. \tag{38}$$

In the infinite time limit (equation 36), the hidden unit converges to the standardised sum,

$$h(x) \to \sqrt{s_n}\, \operatorname{sgn}(\pi_+^0)\, \frac{S - \mathbb{E}\, S}{\operatorname{sd}(S)}. \tag{39}$$

Non-overflow output digits. The digit logit for any position $i \geq 1$, i.e. other than the overflow, at value k, is

$$[\hat{y}]_{i,k}(x) = [W_2 W_1 x]_{i,k} = w_{2\perp}(t)\, \hat{f}_{i,k}\, h(x). \tag{40}$$

In the long time limit from equation 36, $w_{2\perp}(t) \to 0$ while $h(x) \to \sqrt{s_n}\, \operatorname{sgn}(\pi_+^0)\, (S - \mathbb{E}\, S)/\operatorname{sd}(S)$ stays bounded for every x, so

$$\lim_{t\to\infty} [\hat{y}]_{i,k}(x) = 0 \qquad \text{for every } x \text{ and every } k. \tag{41}$$

The model thus fails to learn any output outside the overflow.

Overflow output. The overflow logit is given by $[\hat{y}]_{\text{ovf}}(x) = e_{\text{ovf}}^\top W_2 W_1 x = w_{2\|}(t)\, h(x)$. Plugging this into the decomposition of $h(x)$ from equation 38, we get

$$[\hat{y}]_{\text{ovf}}(x) = w_{2\|}(t) w_{1\|}(t)\, \frac{S - \mathbb{E}\, S}{\operatorname{sd}(S)} + w_{2\|}(t)\, w_{1\perp}(t)\, \hat{e}^\top x. \tag{42}$$

By equation 36, in the infinite time limit, the sign of this becomes

$$\operatorname{sgn}\Big(\lim_{t\to\infty} [\hat{y}]_{\text{ovf}}(x)\Big) \;=\; \operatorname{sgn}\big(S - \mathbb{E}\, S\big) \;=\; \operatorname{sgn}\big(S - (b^n - 1)\big) \;=\; 2y_{\text{ovf}} - 1, \tag{43}$$

for any $S \neq \mathbb{E}\, S$. Unlike the other outputs, the overflow is thus correctly learned.

Removing the overflow. Suppose we drop y_{ovf} from the target, keeping only the digit outputs $(y_1, \ldots, y_n)$. By equation 11, every remaining row of y is independent of x, so

$$\Sigma_{yx} = 0. \tag{44}$$

The dynamics equation 7 become weight collapse, $\dot{W}_1 = -\|W_2\|^2 W_1$, $\dot{W}_2 = -\|W_1\|^2 W_2$, and for balanced initialisation $\rho = \|W_1\| = \|W_2\|$ this gives

$$\tfrac{d}{dt}\rho^2 = -2\rho^4 \quad \Longrightarrow \quad \rho^2(t) = \frac{(\rho^0)^2}{1 + 2(\rho^0)^2 t} \to 0. \tag{45}$$

So $h(x) = W_1 x \to 0$ for every x. Without the overflow, the magnitude representation is not learned.

Encoding invariance of the results. We used a scalar encoding for y_{ovf} to simplify the calculations. The results here are, however, not an artefact of this choice of encoding. Any linear encoding of the overflow $[\tilde{y}]_{\text{ovf}} = \alpha\, (y_{\text{ovf}} - \mathbb{E}\, y_{\text{ovf}})$, where $\alpha \in \mathbb{R}^m$ could be used (e.g. a one-hot encoding corresponds to $\alpha = (-1, 1)^\top$). The overflow row of Σ_{yx} and block of Σ_{yy} gain an extra α term

$$\Sigma_{\tilde{y}x}^{\text{ovf}} = s_n\, \alpha v^\top, \qquad \Sigma_{\tilde{y}\tilde{y}}^{\text{ovf}} = \alpha\alpha^\top \operatorname{Var} y_{\text{ovf}}, \qquad \operatorname{tr}\big(\Sigma_{\tilde{y}x}\Sigma_{x\tilde{y}}\big) = \|\alpha\|^2 s_n^2, \tag{46}$$

which means that all results are either left unchanged or rescaled by the scalar $\|\alpha\|$:

$$\begin{aligned} &v \mapsto v, \qquad \operatorname{sgn}(S - \mathbb{E}\, S) \mapsto \operatorname{sgn}(S - \mathbb{E}\, S), \qquad \frac{s_n^2}{\operatorname{Var} y_{\text{ovf}}} \mapsto \frac{s_n^2}{\operatorname{Var} y_{\text{ovf}}}, \\ &s_n \mapsto \|\alpha\|\, s_n, \qquad \lim_{t\to\infty} \rho = \sqrt{s_n} \;\mapsto\; \sqrt{\|\alpha\|\, s_n}, \qquad t_i \mapsto \frac{1}{\|\alpha\|}\Big(t_i + \frac{\ln \|\alpha\|}{s_n}\Big), \end{aligned} \tag{47}$$

where t_i is the digit resolution time of Section B.4, and $L_\infty > 0$ still holds.

B.4 The decoder

Decoder. The decoder is defined by inverting the part of the hidden representation that encodes the sum, i.e. the first term in equation equation 38:

$$\hat{S} := \mathbb{E}\, S + \operatorname{sd}(S)\, \frac{h}{w_{1\|}}, \qquad \hat{y}_i := \Big\lfloor \frac{\hat{S} + \frac{1}{2}}{b^{\,n-i}} \Big\rfloor \bmod b. \tag{48}$$

Error. For the part of the input orthogonal to the magnitude representation $\eta := \hat{e}^\top x$, we have

$$\mathbb{E}\eta = 0, \qquad \mathrm{Var}(\eta) = \hat{e}^\top \Pi \hat{e} = 1, \qquad \mathrm{Cov}(\eta, S) = \mathrm{sd}(S)\, \hat{e}^\top \Pi v = \mathrm{sd}(S)\, \hat{e}^\top v = 0. \tag{49}$$

Let θ be the angle between $W_1^\top$ and v, so $\cos\theta = w_{1\parallel}/\rho$, $\sin\theta = w_{1\perp}/\rho$ by equation 26, and therefore

$$\tan\theta = \frac{w_{1\perp}}{w_{1\parallel}} = \frac{w_{1\perp}^0 U}{\frac{1}{2}(\pi_+^0 e^{\,s_n t} + \pi_-^0 e^{-s_n t})U} = \frac{2\, w_{1\perp}^0}{\pi_+^0 e^{\,s_n t} + \pi_-^0 e^{-s_n t}}, \tag{50}$$

using equation 31. Plugging the expression for the hidden representation during training from equation 38 into equation 48 gives us the error

$$\Delta := \hat{S} - S = \mathbb{E}\, S + \mathrm{sd}(S)\frac{h}{w_{1\parallel}} - S = \mathrm{sd}(S)\, \tan\theta\, \eta, \tag{51}$$

with

$$\mathrm{sd}(\Delta) = \mathrm{sd}(S)\, |\tan\theta|\, \mathrm{sd}(\eta) = b^n s_n\, |\tan\theta|. \tag{52}$$

Digit resolution times. The digit y_i flips whenever the error moves S across a multiple of b^{n-i}, so digit i begins to resolve once $\mathrm{sd}(\Delta)$ drops to b^{n-i}. Therefore, we define t_i by $\mathrm{sd}(\Delta)(t_i) = b^{n-i}$. Solving for this, i.e.

$$|\pi_+^0 e^{\,s_n t} + \pi_-^0 e^{-s_n t}| = 2|w_{1\perp}^0|\, s_n\, b^{\,i}, \tag{53}$$

gives the solution

$$t_i = \frac{1}{s_n}\Big[i \ln b + \ln \frac{2 s_n\, |w_{1\perp}^0|}{|\pi_+^0|}\Big] - \frac{1}{s_n} \ln\Big|1 + \frac{\pi_-^0}{\pi_+^0} e^{-2 s_n t_i}\Big|, \tag{54}$$

with the last term $O(e^{-2 s_n t_i})$. Equation equation 53 can have two solutions, but only the later one in training is meaningful. Early in training, while the decaying mode $\pi_-^0 e^{-s_n t}$ still dominates $w_{1\parallel}$, the error can transiently go below threshold by a coincidence of the initialisation, before rising back above it. Only for the later solution does the error stay small.

C Additional Experiments

C.1 Digit-by-digit Learning

Acquisition order across seeds. We repeated the exact experimental setup from figure 1 for 10 different seeds, to determine if the effect is consistent across seeds. All of the 10 seeds showed a most-to-least significant learning order for the digits (figure 9).

Staircase. Consistent with (Abbe et al., 2021), we find that the model starts learning each digit only after the digit left of it has finished learning, which can be most easily observed by plotting the digit accuracies relative to the timepoint the previous digit has been learned (figure 10).

Learning order across tasks. In figure 11 the learning dynamics can be seen for the different tasks. The learned digit order flips naturally when removing the overflow, after seeding from below, and is forced to flip by the curriculum. The unseeded run without overflow was cut off after 250k steps, and it cannot be ruled out that it would eventually converge, although not a single digit was learned at that point. Note also that the loss is over the newly generated batch and will generally be unseen for 20-digit addition.

C.2 Mechanistic Analysis

Attention patterns across layers. Figures in the main text show the mechanistic analysis at the final layer of the model, since this layer showed the most structure and is closest to the prediction. In figure 12 attention shares for all layers are shown and in figures 13, 14 and 15, full attention patterns at convergence for all heads are shown for the standard 20-digit, no overflow, and curriculum run respectively.

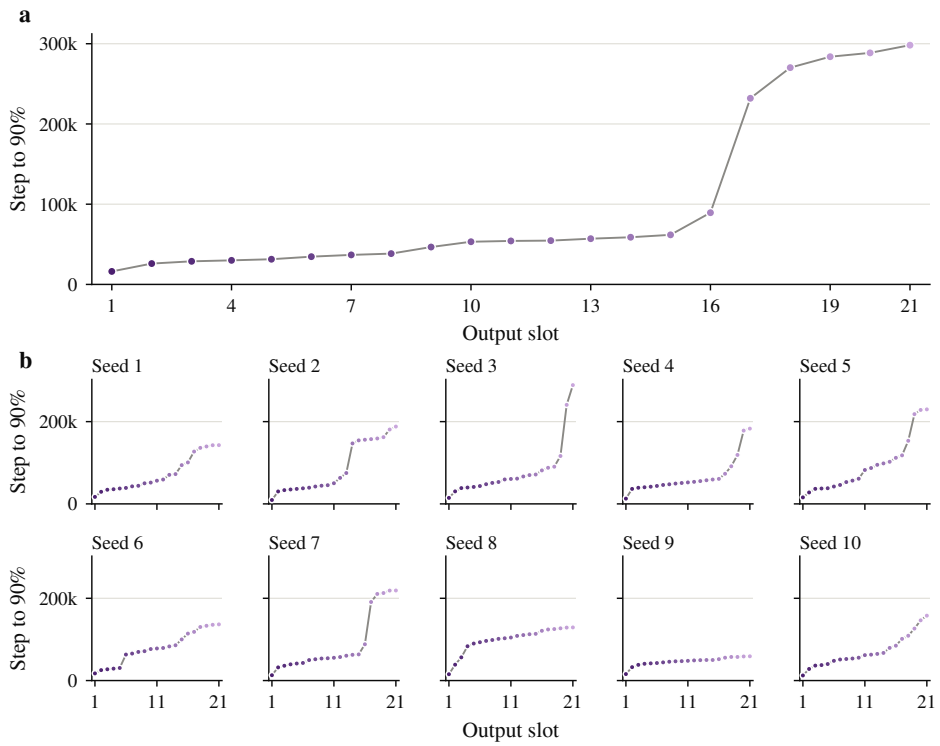


Figure 9: Number of steps to reach 90% accuracy at each output digit position of the encoder transformer trained directly on 20-digit addition, for **(a)** the seed from figure 1 and **(b)** 10 other seeds.

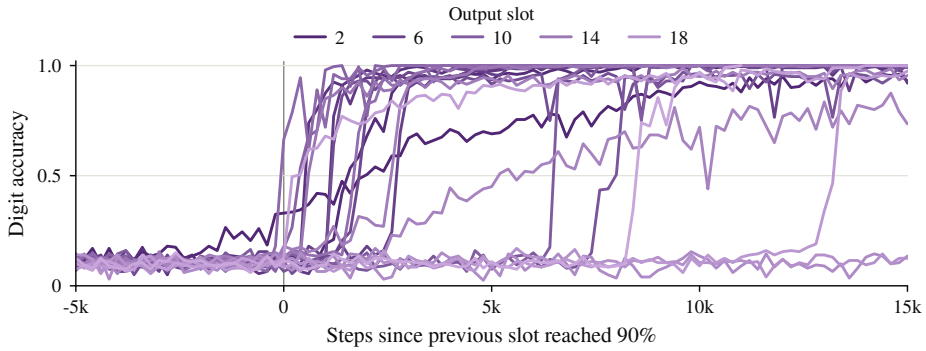


Figure 10: Accuracy per digit for the encoder transformer trained on 20-digit addition, as a function of the time since the digit left of it has reached 90% accuracy.

Head patching. Just because a head has an attention pattern consistent with a standard diagonal or carry computation, does not mean it is actually performing this computation. To understand better the computation the discovered heads actually perform, we use activation patching (Vig et al., 2020; Meng et al., 2022). First we group heads based on their attention patterns, by scoring them on how much attention falls either on the diagonal, or the position-below (figure 16a). Next we generate minimal edit pairs, for both the diagonal and carry, that change the correct value at a specific output position. The diagonal edits are performed by changing one of the input digits at the same position as the scored output digit, causing the output digit to change. The carry edits instead change the output

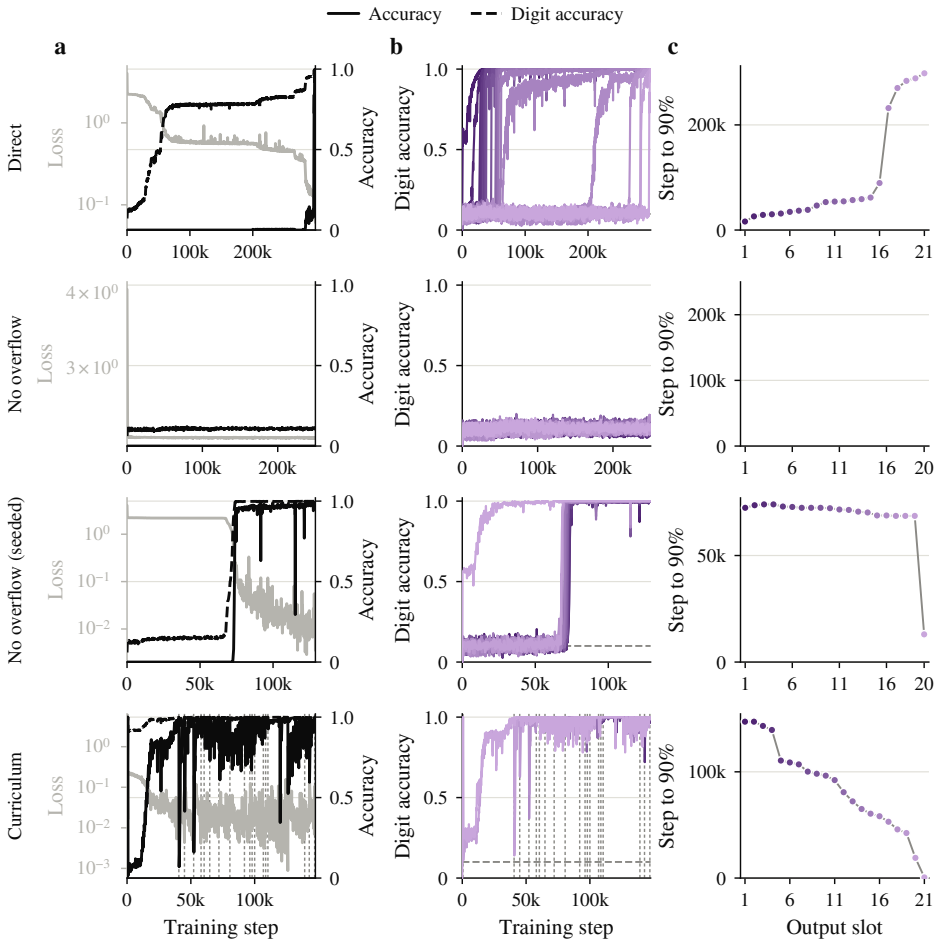


Figure 11: Training dynamics for the model trained on 20-digit addition, a version where the overflow is removed, a version where the overflow is removed and the least significant digit replaced by its binary carry value, and a curriculum of increasing digit counts. **(a)** Loss, exact-match accuracy, and average digit accuracy during training. **(b)** Digit-wise accuracy during training. **(c)** Training steps to reach 90% accuracy per digit.

digit by editing an input digit at one position lower, causing a change in carry. For example:

Own position	clean	$1\underline{2}53 + 6344$	$=$	$7\mathbf{5}97$
	edited	$1\underline{6}53 + 6344$	$=$	$7\mathbf{9}97$
Position below	clean	$1253 + 63\underline{4}4$	$=$	$7\mathbf{5}97$
	edited	$1253 + 63\underline{5}4$	$=$	$7\mathbf{6}07$

We first run the model on both the clean and edited datapoint. Then, for datapoints where the model got both correct, we do a run where the activations from the clean run are patched into the edited run, for all heads in one of the three groups, to see if this flips the targeted output digit towards that of the clean datapoint. We find for both the model trained without the overflow and the model trained with the curriculum, that the diagonal attention pattern group flips only on the diagonal edits, the position-below group flips only on the carry edits, and the remaining heads do not cause a flip on either (figure 16b). For the model trained directly on 20-digit addition, none of the heads showed structure.

Misleading problems for estimation. A behavioural test for an estimation strategy is to look only at the accuracy on problems which are misleading for such a strategy. For example, consider the

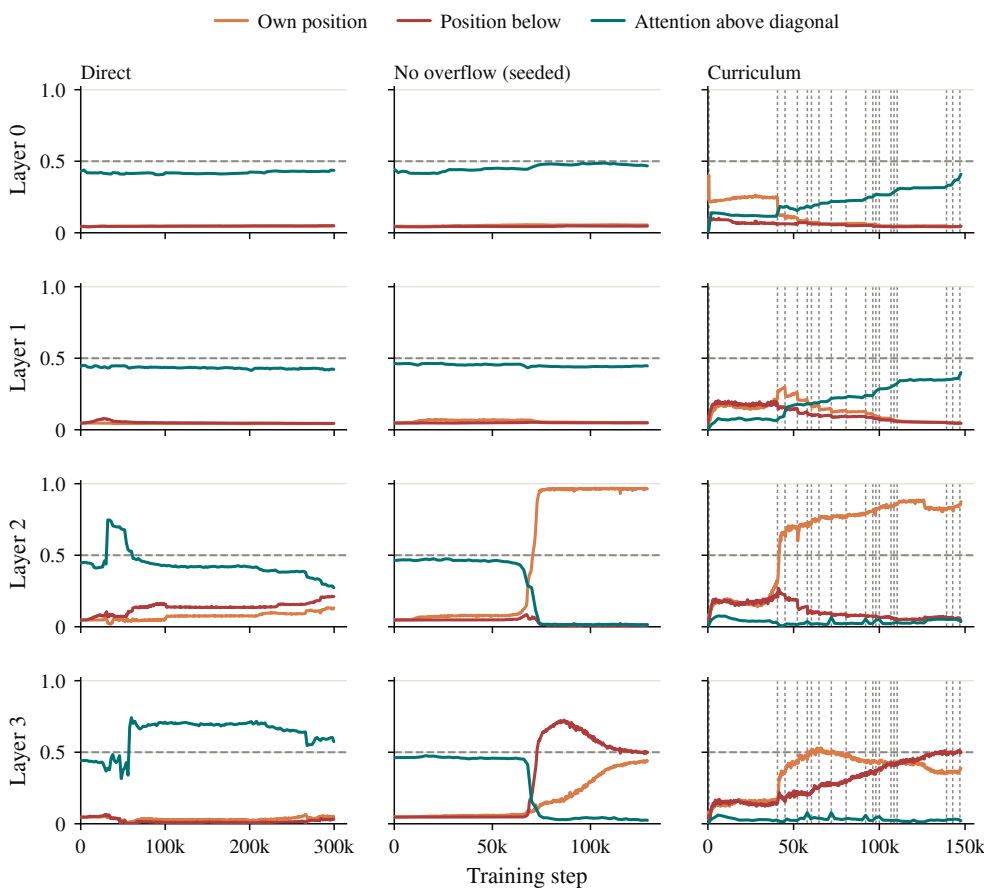


Figure 12: Attention share averaged during training for each layer averaged over heads, on the diagonal (orange), on one position below where a carry computation would need to look (red), and positions above the diagonal (teal). Results are shown for the model trained directly on 20-digit addition, the model trained without the overflow digit and the least significant digit replaced by its carry value, and the model trained with a curriculum of increasing lengths.

following addition problem:

$$\begin{array}{r} 1096 \\ +2098 \\ \hline 3194 \end{array}$$

A digit-wise algorithm would easily be able to tell here that the most significant output is 3, since at the position before we have two zeros. However, the less significant digits overall are very high, so an estimation strategy will be likely to guess that there is a carry, since this is statistically very likely based only on which digits are present. We generate 500 of such problems, and score the model when trained on each of the three tasks. Even though the model has a similar accuracy on all three settings, we see that only the model trained directly on addition performs very poorly on these problems (figure 17).

C.3 Additional Transfer and Generalisation Results

Transfer null comparison. To understand whether the fine-tuning benefits of the increasing length curriculum are related specifically to the model learning the digits from right to left or a general property of learning with a curriculum, we compare it to a different curriculum. To make sure this curriculum is unrelated to digit learning order, we instead apply a curriculum to the digit alphabet, meaning the model starts training on numbers only containing zeros, then numbers with only zeros

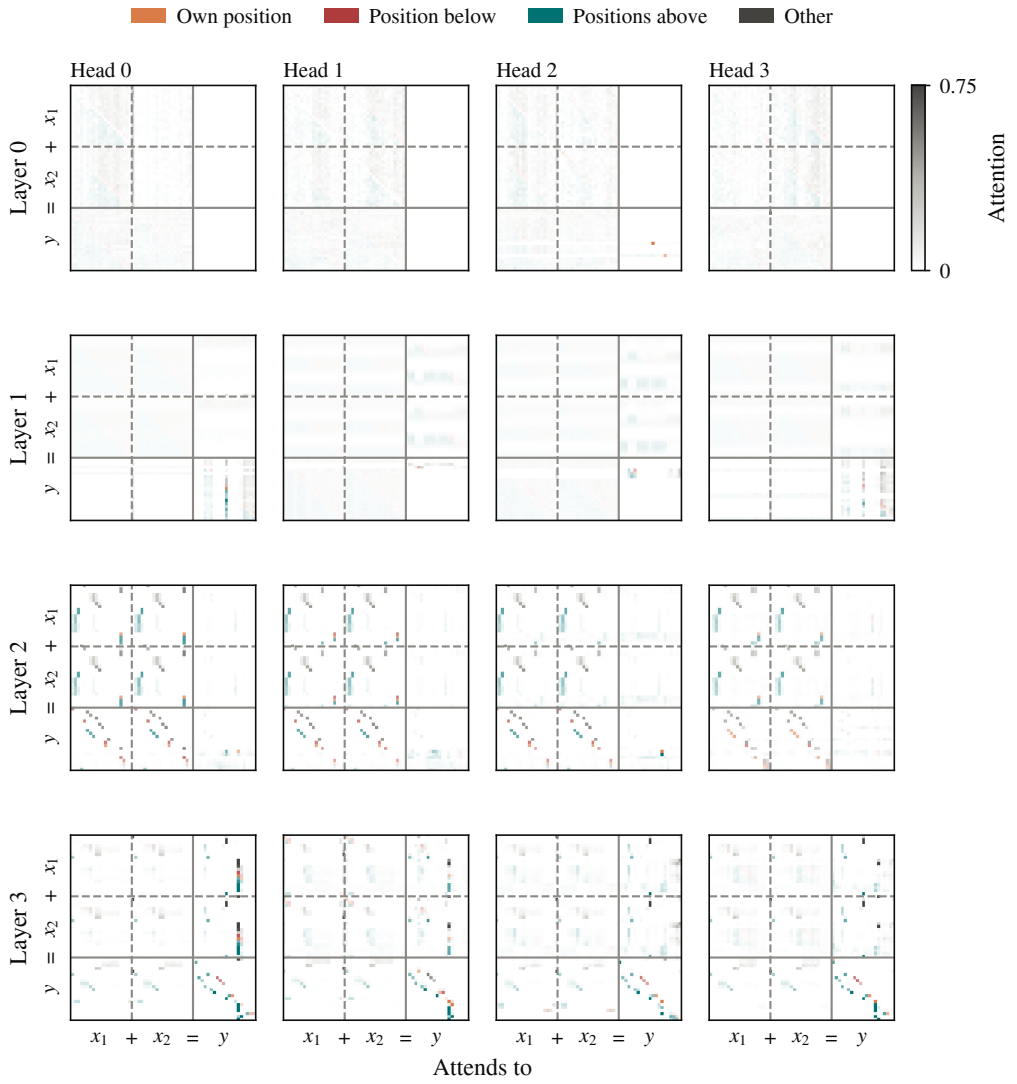


Figure 13: Attention patterns averaged over 100 datapoints for every head from the encoder model after training directly on 20-digit addition. The diagonal is coloured orange, one position below the diagonal where a carry computation would need to look is coloured red, and positions above the diagonal coloured teal. No heads match a typical digit-wise addition computation (diagonal or one-below diagonal).

and ones, and so on. In figure 18 we can see that this curriculum has some benefit for the base-7 task, but shows no improvement on the other two tasks compared to the model trained without curriculum.

Training time. The increasing length curriculum model converged faster on the 20-digit task (figure 19). Additionally, the training time benefit appears to increase with the number of digits, e.g. showing virtually no improvement in training time below 18 digits and more than a doubling of training speed at 40 digits (figure 20). These results were not observed for the digit alphabet curriculum.

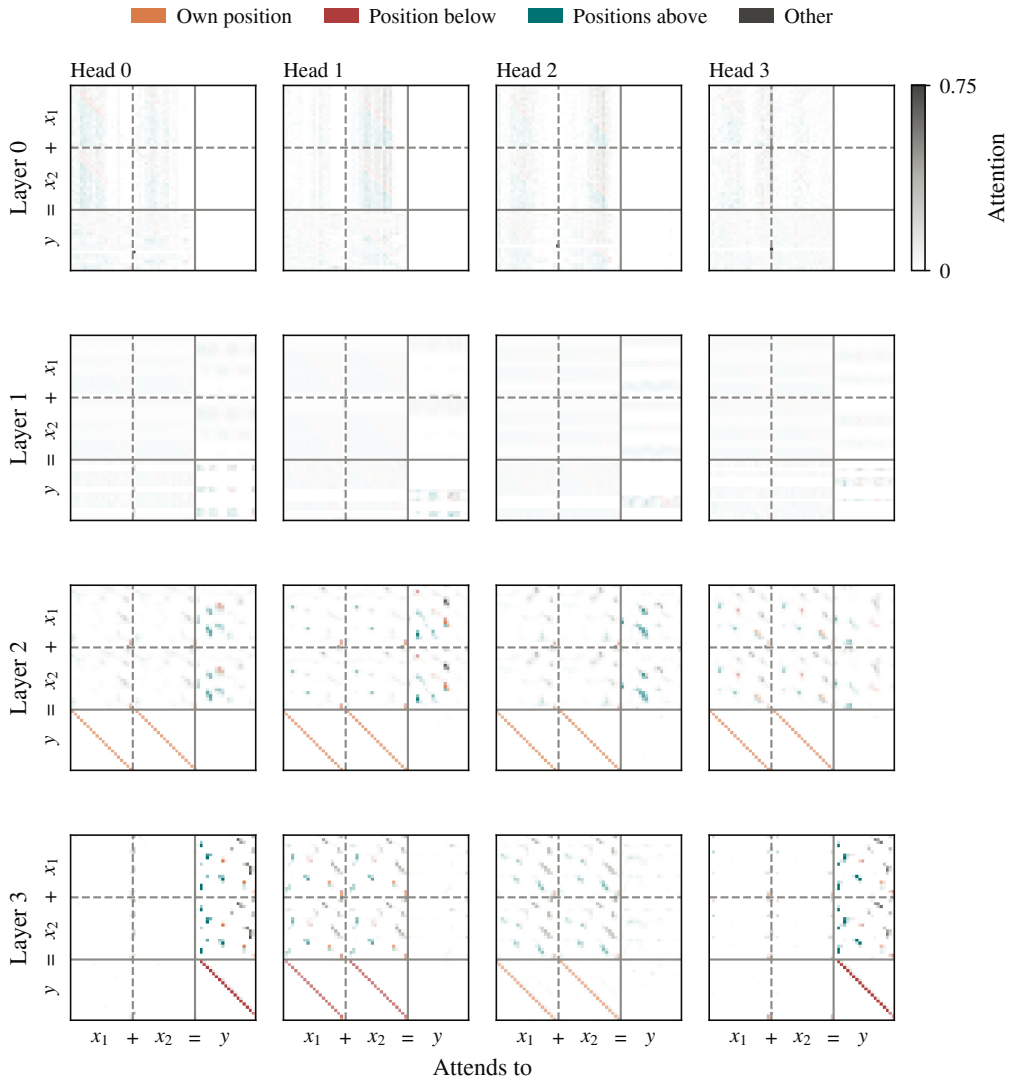


Figure 14: Attention patterns averaged over 100 datapoints for every head from the encoder model after training on 20-digit addition without the overflow digit, seeded from below. The diagonal is coloured orange, one position below the diagonal where a carry computation would need to look is coloured red, and positions above the diagonal coloured teal. Some heads match a typical digit-wise addition computation in the third and fourth layer.

C.4 Additional Settings

Finite data. In figure 21 the training dynamics and head classification can be seen as a function of dataset size. We observe a regime where generalisation accuracy increases only long after the model fits the data. Carry heads only emerge at a larger dataset size.

Architectures. In figure 22 results are shown for different architectures: the transformer encoder from the main text, an identical transformer but with causal masking to make it auto-regressive, a Bahdanau-style recurrent neural network consisting of single layer GRU encoder and decoder with an attention layer in between and hidden dimension 256, and a 4-layer ReLU MLP with hidden dimension 2048. For the autoregressive transformer the digit-learning order matches the generation order. The RNN appears to learn digits from both sides at the same time. The MLP, which just

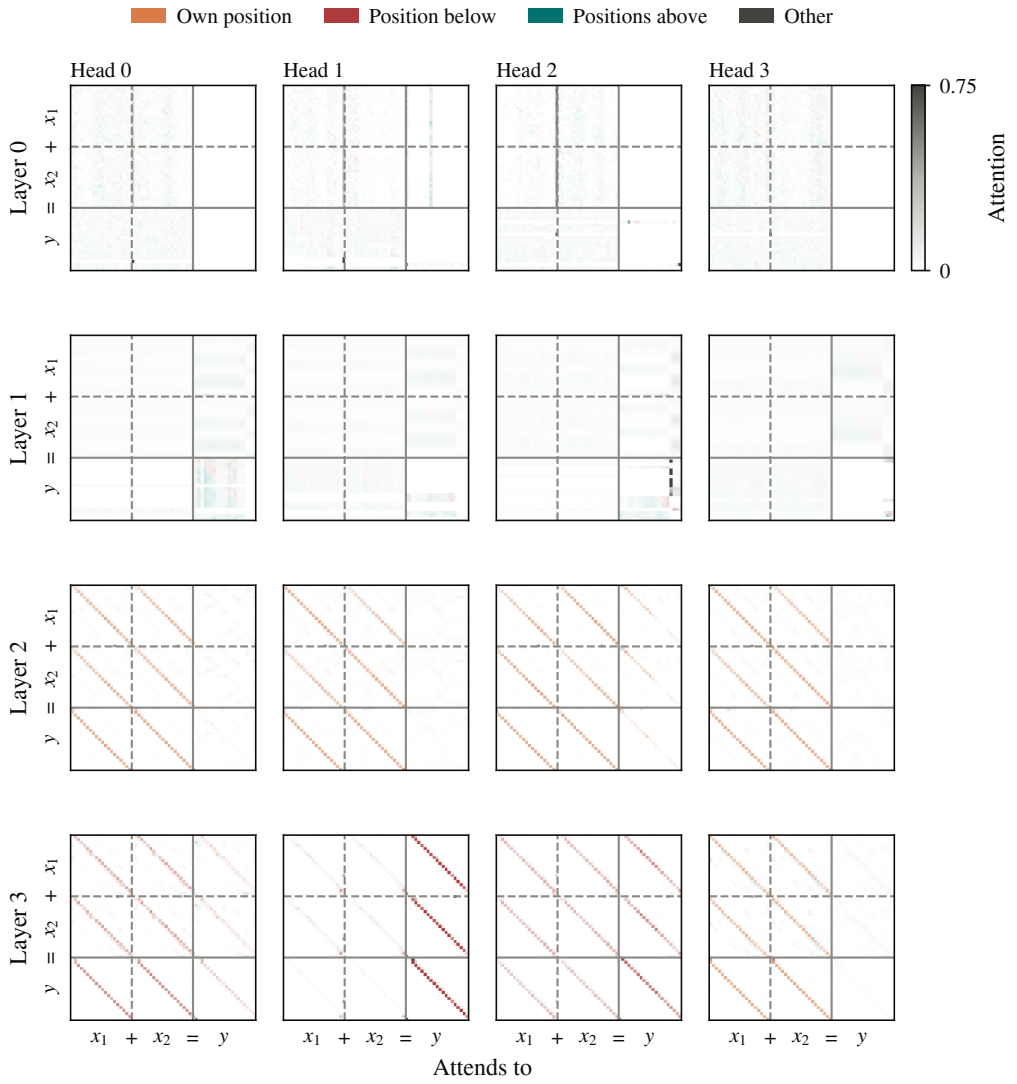


Figure 15: Attention patterns averaged over 100 datapoints for every head from the encoder model after training on 20-digit addition with an increasing digit size curriculum. The diagonal is coloured orange, one position below the diagonal where a carry computation would need to look is coloured red, and positions above the diagonal coloured teal. Most heads match a typical digit-wise addition computation in the third and fourth layer.

like the non-autoregressive transformer does not have a forced digit order, also learns digits from most significant to least. The autoregressive transformer and RNN both show high accuracy on any carry-chain, when fed the sequence from least to most significant digits. The MLP only displayed a smooth digit acquisition order curve at high width and low learning rate.

Different tasks. Both subtraction and binary addition show a similar pattern, where the non-autoregressive transformer learns digits from most significant to least (figure 23). In both cases, the increasing length curriculum speeds up learning.

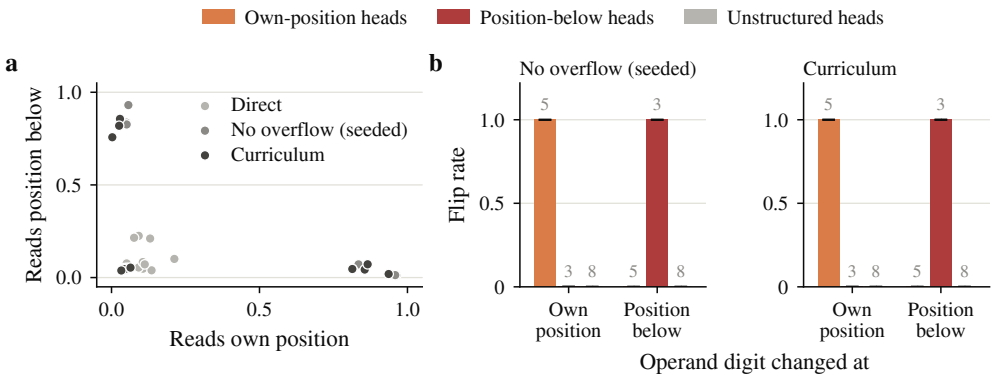


Figure 16: **(a)** Heads plotted by diagonal and position-below attention fractions. They form three clusters: heads which attend on the diagonal, heads which attend at the position below, and unstructured heads which do neither. **(b)** Flip rates after patching on edits at inputs at the output position, or edits at the position below. Patching was performed on all heads in one of the three groups. The diagonal pattern heads cause a flip only for the edit at the same position, the position-below pattern heads only for the position-below edit, and the remaining heads for neither edit.

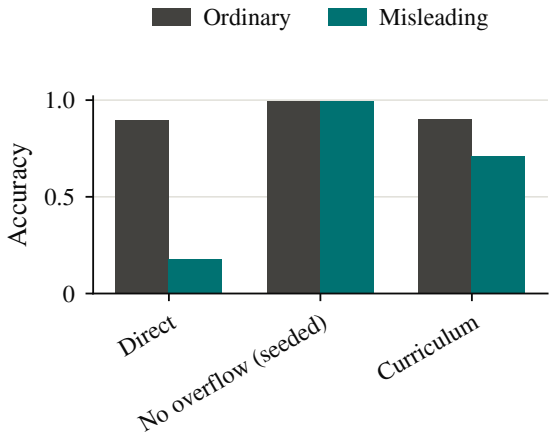


Figure 17: Accuracy of standard addition problems and problems misleading for estimation strategies. Note that only the model trained directly on 20-digit addition shows a near zero score on the misleading problems.

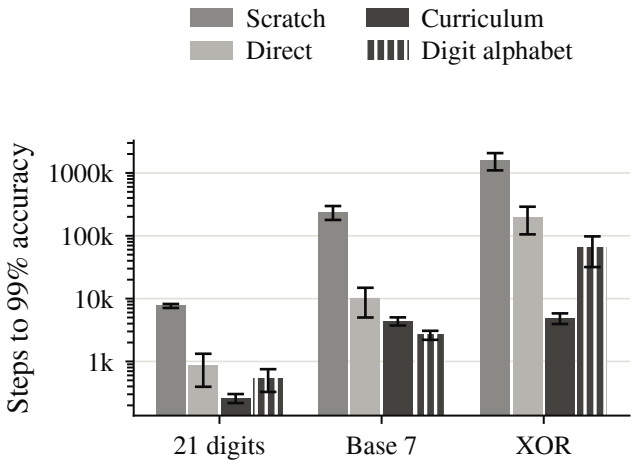


Figure 18: Number of training steps to 99% accuracy on 21-digit addition, 20-digit addition in base 7, and 20-digit position-wise XOR, where the model is either trained from scratch, or fine-tuned from the direct, increasing length curriculum model, or increasing digit alphabet curriculum model. Note that the digit alphabet model has increased transfer performance on base 7, but not on the other two tasks compared to the directly trained model.

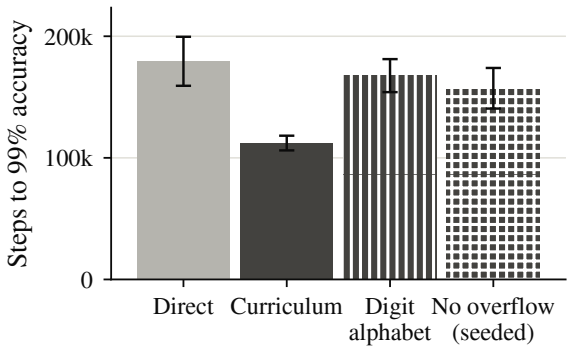


Figure 19: Number of training steps until 99% accuracy, for the model trained directly on 20-digit addition, trained with the increasing length curriculum, trained with the digit alphabet curriculum, and trained on the task without the overflow, seeded at the ones position. Results have been averaged over 10 seeds. Note a decrease in training time with the increasing length curriculum.

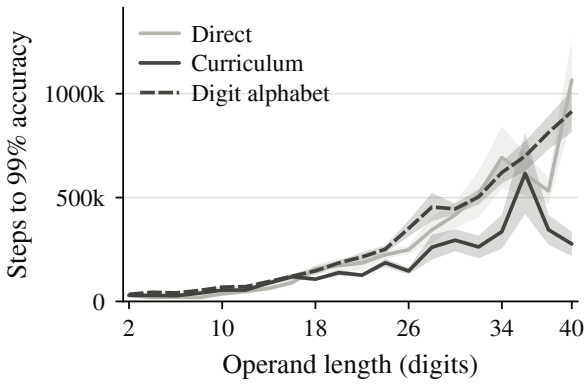


Figure 20: Number of training steps until 99% accuracy, as a function of the number of digits in the task. Results have been averaged over 20 seeds. Note that especially for longer number lengths with the curriculum the model learns the task faster.

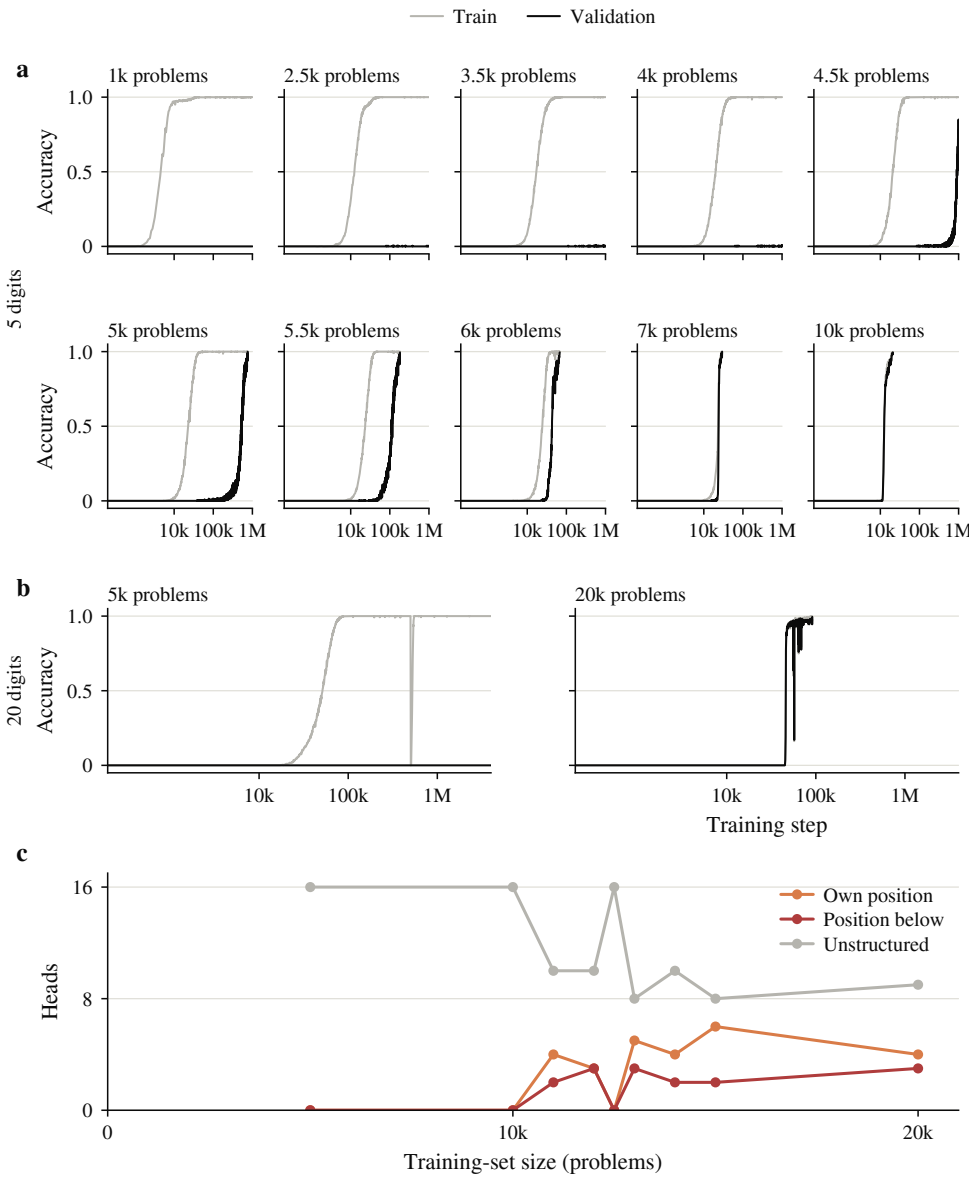


Figure 21: **(a)** Training and validation accuracy for the encoder transformer trained on 5-digit addition, with finite datasets of varying sizes. A transition from overfitting to generalisation can be observed, with a regime in the middle where generalisation occurs much later than fitting the training data. **(b)** The model trained on 20-digit addition instead. The transition from the overfitting to generalising regime becomes much more narrow. **(c)** Classification of heads based on attention patterns of the model trained on 20-digit addition as a function of dataset size. Diagonal and carry heads emerge at larger dataset sizes.

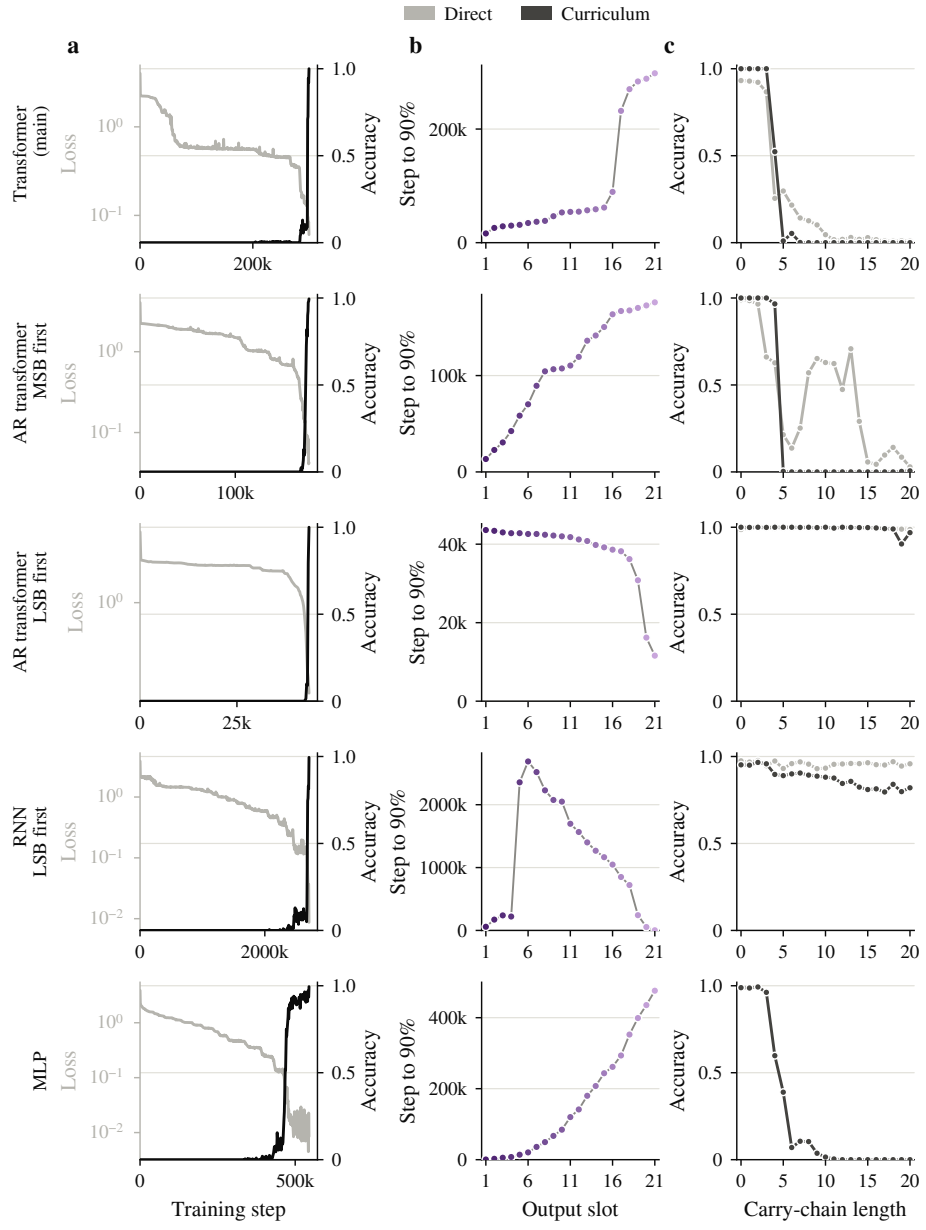


Figure 22: Comparison of training dynamics across the encoder transformer from the main text, a decoder transformer being fed either most significant digits, or least significant digits first, a Bahdanau-style recurrent neural network fed least significant digits first, and an MLP. The recurrent neural network when fed digits from most significant to least significant did not converge. **(a)** Loss and accuracy during training. **(b)** Training steps to reach 90% accuracy. **(c)** Accuracy on numbers containing different carry-chain lengths.

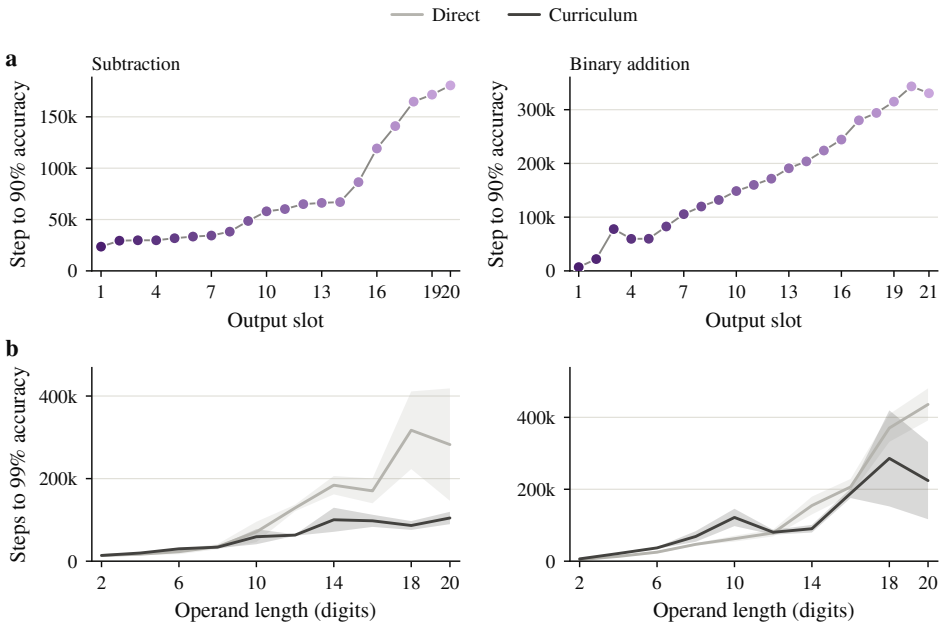


Figure 23: **(a)** Training steps to reach 90% accuracy per digit in subtraction and binary addition, when the encoder transformer is trained directly on them. Both show the reverse digit learning pattern. **(b)** Number of training steps until 99% accuracy, as a function of the number of digits in the task. Results have been averaged over 10 seeds. For longer numbers the increasing length curriculum speeds up learning on both tasks.