Learning Large-Scale Modular Addition with an Auxiliary Modulus

Hanato Kikuchi
Chiba University, Chiba, Japan
hanato.kikuchi@chiba-u.jp

Ryosuke Masuya
Chiba University, Chiba, Japan
ryosuke_masuya@chiba-u.jp

Kazuhiko Kawamoto
Chiba University, Chiba, Japan
kawa@faculty.chiba-u.jp

Hiroshi Kera*
Chiba University, Chiba, Japan
National Institute of Informatics, Tokyo, Japan
kera@chiba-u.jp

Abstract

Learning parity functions, or more generally modular addition, is a challenging machine learning task due to its input sensitivity. A recent study substantially scaled modular addition learning in both the number of summands and the modulus. Its key idea is to increase zeros in training sequences, reducing the effective number of summands and thus controlling training difficulty; however, this induces covariate shift between training and test input distributions. This study theoretically and empirically analyzes this side effect and proposes a covariate-shift-free method for modular addition. Specifically, we introduce an auxiliary modulus Kq during training, which reduces wrap-around frequency and problem difficulty while preserving the same input distribution across training and testing. Experiments show strong scalability and sample efficiency: even for large input length N, large modulus q, and small datasets—where the sparse method fails to learn—our method achieves equal or better match accuracy and relaxed τ-accuracy. For example, at $N = 128$ and $q = 974269$, our method trained on 1M samples achieves 97.0% τ-accuracy at $\tau = 0.005$, while the sparse method achieves only 9.5% with the same data size and 93.9% even when extended to 10M samples.[2]

1 Introduction

Despite recent success in learning complex symbolic computational tasks, such as symbolic integration [6] and Gröbner basis computation [13], and so on [2, 14], learning modular addition is still a challenging task for deep learning models.

Definition 1 (Modular Addition)**.** *Let $N, q \in \mathbb{N}$ and $q \geq 2$. Given a sequence $\mathbf{x} = [x_1, \ldots, x_N]$ with $x_i \in \{0, \ldots, q-1\}$, compute the sum $\sum_{i=1}^{N} x_i$ modulo q, i.e., $f_q(\mathbf{x}) := \sum_{i=1}^{N} x_i \mod q$.*

The difficulty of learning f_q arises from the input sensitivity: even a single change in the input can significantly affect the output. Theoretical analysis by Shalev-Shwartz et al. [25] and Hahn and Rofin [11] has proved this for the parity problem (i.e., modular addition with $q = 2$). Larger N and q increase the difficulty further as it incurs more wraps around the modulus. The number of required training samples grows with N and q [20]; besides, large N necessitates an increased network width [8].

*Corresponding author.

[2]Code is available at https://github.com/kawakera-lab/auxiliary_modulus.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

A recent study by Saxena et al. [24] identified learning modular addition as a key component of attacking the Learning With Errors (LWE) problem. LWE is a basis of major post-quantum cryptosystems [27]. For a binary secret key, recovering the secret key from noisy modular linear equations reduces to modular addition, so modular addition plays an important role in machine-learning attacks on LWE. The study proposed a method that substantially scaled modular addition in both the number of summands and the modulus. Particularly, addressing large modulus q is crucial for their application. A naive training even leads to stagnated training loss because of the small variance of the gradient of the loss function [25]. Thus, the key is to limit wrap-around frequency at training time and ease the problem difficulty to make training flow. Saxena et al. [24] implemented this idea by increasing zeros in training sequences, thereby lowering the effective number of summands; however, this induces covariate shift between training and test input distributions, which is unfavorable from a standard machine learning perspective.

In this study, we propose a new learning method for modular addition that involves no covariate shift and attains scalable and sample-efficient learning. The proposed method introduces an auxiliary loss of modular addition with enlarged modulus Kq for $K > 1$. The enlarged modulus reduces wrap-around frequency and problem difficulty at training time, while preserving the input distribution for both training and testing. Our experiments demonstrate that elimination of covariate shift substantially improves the sample efficiency and scalability of modular addition learning: the proposed method achieves competitive or superior accuracy compared to the sparse method by Saxena et al. [24] under various conditions of N, q, and even with a smaller training sample size.

We summarize the contributions as follows.

- We examine the sparse method by Saxena et al. [24] and theoretically and empirically demonstrate the side effect of covariate shift. Specifically, training on sparse inputs with more zeros than dense inputs in the test distribution induces non-trivial generalization error. Besides, the success of the sparse method is sensitive to the model configuration, such as Dropout, PreNorm, Bias, and weight initialization.
- We propose a covariate-shift-free method for modular addition learning. An auxiliary loss of modular addition with enlarged modulus Kq for $K > 1$ is introduced. This loss replaces the main loss with probability r. The enlarged modulus reduces wrap-around frequency and problem difficulty at training time, while preserving the input distribution for both training and testing.
- Our method scales to larger problem sizes beyond the reach of the prior method. Concretely, at $N = 128$ and $q = 974269$, training on only 1M samples yields 97.0% τ-accuracy (with relative error tolerance $\tau = 0.005$), whereas the sparse method attains 9.5% at the same scale and only 93.9% even with 10M samples.

2 Related Work

We summarize related work on learning modular arithmetic using deep learning models, methods for utilizing easier examples, and auxiliary tasks.

Learning modular addition with deep learning models. Recent studies have investigated how deep learning models learn modular addition. These works show that models learn this task by acquiring internal periodic structures. For instance, this periodicity is confirmed in algorithm models learned for modular addition. Specifically, it has been shown that models treat numbers as points in Fourier space and learn modular addition through algorithms such as addition of angles [21], the "Pizza" algorithm [30], and the approximate Chinese Remainder Theorem [19]. Additionally, the weights and embeddings of models trained on modular addition also exhibit periodic structures [17, 22], and periodic analytical solutions for these weights have been discovered [10]. Furthermore, these periodic structures are observed in other modular arithmetic tasks, such as modular multiplication [8, 9]. Therefore, helping models learn periodic structures is important. Previous methods, such as angle embeddings [26], directly add periodicity by treating numbers as points on a circle. Instead of changing the inputs, our method uses an auxiliary task with a different period (modulus Kq) to help the model learn periodicity.

Learning from simpler tasks. Utilizing simpler tasks is effective for learning modular addition and parity problems. For instance, one approach uses Chain of Thought [28] to decompose problems into steps, treating them as simpler sub-problems. This enables models to solve parity problems that are otherwise difficult to learn [15, 29]. Another effective approach is curriculum learning [3], which gradually increases task difficulty. For parity problems, sparse parity problems [7] are utilized as simpler tasks, and applying curriculum learning with them improves both learning accuracy and efficiency on the target problem [1]. Similarly, for modular addition, recent work mixes sparse and standard tasks during training, successfully learning more complex modular addition (e.g., up to $N = 128$ terms and modulus $q = 974269$) than conventional methods [24]. However, the method in Saxena et al. [24] causes covariate shift because mixing easier problems only during training alters the input distribution (as detailed in Section 3). In contrast, our method avoids this by using an auxiliary task to learn both problems simultaneously, keeping the input distribution consistent.

Learning with auxiliary tasks. Auxiliary task learning improves generalization [5, 23]. In mathematical tasks, learning different operations simultaneously improves accuracy. For example, joint training on multiple arithmetic operations enhances performance [16]. Furthermore, combining simple and difficult tasks reduces model size if they share core building blocks [4]. While Both et al. [4] simultaneously train on different operations (e.g., modular addition and modular multiplication with a non-prime modulus), we use a highly related task: the same modular addition with a different modulus.

3 Problem Setting

This study tackles modular addition learning (cf. Definition 1). Concretely, we adopt the setup below.

Training on sparse inputs. A very recent state-of-the-art method for learning modular addition—which we refer to as the sparse method— is to train on inputs that are sparser than samples from the uniform distribution on $\{0, \ldots, q-1\}^N$ [24]. Concretely, the procedure first samples the number of non-zero entries $z \in \{1, \ldots, N\}$ from $g(z) \propto 1/\sqrt{N - z + 1}$, then chooses z positions uniformly at random and fills them independently with values from $\{0, \ldots, q-1\}$, leaving the remaining $N - z$ positions as zero. While g itself peaks at $z = N$ (a fully populated input), it has a much heavier tail toward small z than the uniform marginal $\mathrm{Bin}(N, (q-1)/q)$; hence the resulting samples are typically sparser than uniform ones. Involving more zeros implies fewer wraps around the modulus, thereby reducing the difficulty of the problem.

While the sparse method is empirically effective, it seems unnatural because it introduces a covariate shift between the training distribution (many zeros in $\mathbf{x}$) and the test distribution (uniformly sampled from $\{0, \ldots, q-1\}^N$). To examine this, we quantify the resulting generalization gap.

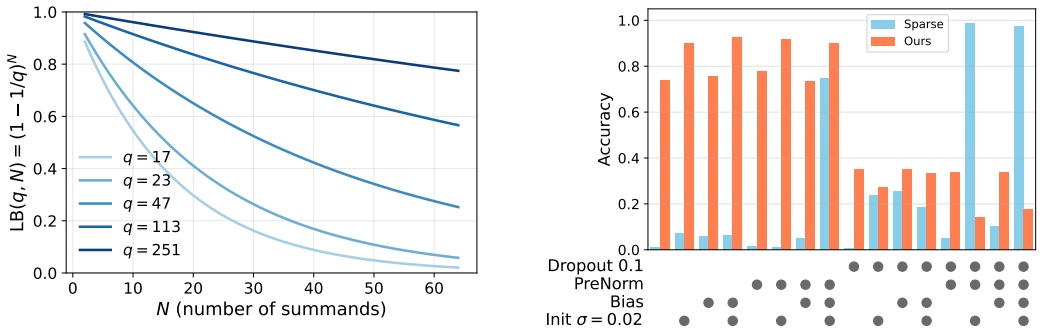
Proposition 1 (Generalization gap under sparse training, informal)**.** *Let P_{te} be the uniform distribution on $\mathcal{X} = \{0, \ldots, q-1\}^N$ and let P_{tr} be the sparse construction of [24]. Write $n_0(x) := \#\{i : x_i = 0\}$ for the number of zero entries, and call an input* zero-free *if $n_0(x) = 0$ (all entries are non-zero). Suppose the model attains training risk $R_{\mathrm{tr}}(f) \le \delta$ but its expected loss on a uniformly random zero-free input is at least ε. Then*

$$R_{\mathrm{te}}(f) - R_{\mathrm{tr}}(f) \;\ge\; \varepsilon\Big(1 - \tfrac{1}{q}\Big)^N - \delta. \tag{1}$$

Proof sketch. Conditioning on $n_0(x)$ gives $R_{\mathrm{te}}(f) \ge P_{\mathrm{te}}(n_0(x) = 0)\,\varepsilon = (1 - 1/q)^N \varepsilon$; subtracting $R_{\mathrm{tr}}(f) \le \delta$ yields Equation (1). See Section A for the formal statement and full proof.

The prefactor $(1 - 1/q)^N$ is large in typical modular-addition regimes (0.70 for $(q, N) = (23, 8)$, 0.93 for $(113, 8)$, 0.75 for $(113, 32)$; see Figure 1a) and tends to 1 as $N/q \to 0$, in which case Equation (1) reduces to $R_{\mathrm{te}}(f) - R_{\mathrm{tr}}(f) \gtrsim \varepsilon - \delta$: any non-trivial error on the zero-free inputs that dominate the test distribution translates into an irreducible test-time gap.

Model configuration sensitivity. Beyond this theoretical bound, we observe empirically that the sparse method is sensitive to the model configuration (see Figure 1b). Specifically, when varying the active components—such as Dropout, PreNorm, Bias, and weight initialization—the performance of the sparse method frequently collapses, often failing to learn entirely. In contrast, the proposed



(a) Generalization gap lower bound vs. N.

(b) Match accuracy vs. model configuration.

Figure 1: **(a)** Lower bound $\mathrm{LB}(q, N) = (1 - 1/q)^N$ of Equation (1) as a function of N for several moduli q. Larger q (darker blue) keeps the bound substantial across the whole range, while for small q the bound vanishes as N grows. See Section A for the underlying table. **(b)** Comparison of match accuracy under various model configurations ($N = 8$, $q = 97$). The lower panel indicates the active components for each configuration; the upper panel compares the sparse method (blue) and ours (orange).

method is highly robust to such structural changes. Further details and comprehensive evaluations are provided in Section B. Although we do not have direct evidence that the covariate shift causes this, it is reasonable to consider that learning under covariate shift requires a careful training setup.

Note that the discussion above should not be read as a criticism of the sparse-input work of [24]. Modular addition is hard enough that, without biasing training towards easier instances, even the training loss fails to decrease within practical compute budgets. Controlling problem difficulty is, therefore, necessary; they already demonstrated one concrete way to do this. What we pinpoint is only a side effect of *how* this control is realized—inserting many zeros creates a covariate shift that bottlenecks the test risk.

Therefore, the desirable direction is to control the difficulty of training problems without introducing a covariate shift; we next present a method that achieves this.

4 Method

We now present our learning pipeline for the modular addition task. An important, well-known fact is that learning addition over $\mathbb{Z}$ is easy while learning addition over $\mathbb{Z}/q\mathbb{Z}$ is hard. That is to say, the difficulty of the problem derives from wrapping around the modulus.

4.1 Auxiliary modulus Kq

The central idea is to reduce the number of wraps around the modulus during training to facilitate learning. Saxena et al. [24] implemented this idea by sparsifying the input $\mathbf{x}$, but this leads to a covariate shift between the training and testing. Instead, we introduce an auxiliary modulus Kq, $(K > 1)$ to reduce the number of wraps around the modulus during training. Formally, with $r \in [0, 1]$, we define the target label y for input $\mathbf{x}$ as follows:

$$y = \begin{cases} f_q(\mathbf{x}) & \text{with probability } 1 - r \\ f_{Kq}(\mathbf{x}) & \text{with probability } r. \end{cases} \tag{2}$$

Namely, we train the model to predict the solution to modular addition with modulus q with probability $1 - r$ and the solution to modular addition with modulus Kq with probability r.

Importantly, the input remains restricted to $\{0, \ldots, q-1\}$, i.e., $\mathbf{x} \in \{0, \ldots, q-1\}^N$, for the auxiliary modulus Kq. The number $D_{Kq}(\mathbf{x})$ of wraps around the (extended) modulus for $\mathbf{x}$ is consequently

reduced to

$$D_{Kq}(\mathbf{x}) = \left\lfloor \frac{\sum_{i=1}^{N} x_i}{Kq} \right\rfloor, \tag{3}$$

where $\lfloor \cdot \rfloor$ denotes the floor function. The expected number of wraps around the modulus is then $\mathbb{E}[(1-r)D_q + rD_{Kq}]$.

4.2 Analysis of Problem Difficulty Reduction

We examine the reduction of the problem difficulty by the proposed method and the sparse method. Here, the problem difficulty is measured by the expected number of wraps around the modulus.

We keep the notation as in Section 4. The floor-free baseline and the floored wrap counts are

$$Y_0 = \frac{\sum_{i=1}^{N} x_i}{q}, \quad X_0 = D_q(\mathbf{x}), \quad X_1 = (1-r)D_q(\mathbf{x}) + rD_{Kq}(\mathbf{x}), \quad X_2 = \left\lfloor \frac{\sum_{i=1}^{z} x_i}{q} \right\rfloor.$$

where z is a random variable that represents the number of non-zero entries of an input data, and has a distribution $f_{\text{inv_sqrt}}(z) \propto 1/\sqrt{N-z+1}$ introduced in Saxena et al. [24], and x_i's are i.i.d. random variables with a discrete distribution $U_q(x) = 1/q$ if $x \in \{0, \ldots, q-1\}$ otherwise 0. The random variable $X_0 = D_q(\mathbf{x})$ is the number of wraps for the naive method without any training difficulty reduction, and Y_0 is that count without the floor function. The random variables X_1 and X_2 are the floored wrap counts of the proposed method and the sparse method, respectively. Then, we have the following proposition:

Proposition 2. *Keep the above notation. For Y_0, we have*

$$\frac{\mathbb{E}[Y_0]}{N} = \frac{q-1}{2q}.$$

Fixed q, K and r, we have

$$\lim_{N\to\infty} \frac{\mathbb{E}[X_0]}{N} = \frac{q-1}{2q},$$
$$\lim_{N\to\infty} \frac{\mathbb{E}[X_1]}{N} = \left((1-r) + r\frac{1}{K}\right)\frac{q-1}{2q},$$
$$\lim_{N\to\infty} \frac{\mathbb{E}[X_2]}{N} = \frac{2}{3}\frac{q-1}{2q}.$$

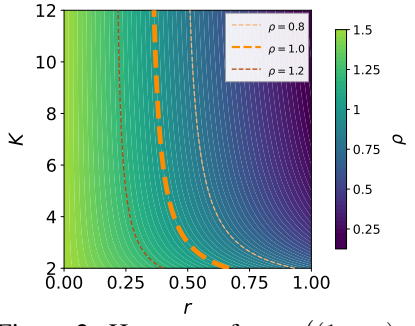


Figure 2: Heatmap of $\rho = \big((1-r) + r/K\big)/(2/3)$ in Equation (4). The orange contour is defined by $\rho = 1$. The range of ρ is from 0.0 to 1.5.

When N is large, Proposition 2 shows that all expectations have the same factor $N(q-1)/2q$ $(= \mathbb{E}[Y_0])$, so the dependence on N and q is identical at this coarse level up to constant multipliers. This indicates that the proposed method realizes a similar reduction of the problem difficulty as the sparse method. Figure 2 shows the ratio

$$\rho = \left((1-r) + r\frac{1}{K}\right) \Big/ \frac{2}{3} \tag{4}$$

of the constant factor between the proposed method and the sparse method over K, r. We observe $\rho \approx 1$ for the combination of K and r that perform well in the experiments.

A full proof of Proposition 2 and additional analysis of the wrap-count variable are given in Section C. Section C also examines detailed properties of D_{Kq}.

4.3 Computational pipeline

We here assume the Transformer model for the explanation clarity, yet our method is not limited to this model. The input $\mathbf{x} = [x_1, \ldots, x_N]$ is tokenized into a sequence of N tokens $[\mathtt{x}_{x_1}, \ldots, \mathtt{x}_{x_N}]$, where $\mathtt{x}_n$ is the token of scalar n. The embedding layer maps each token to an embedding vector. Transformer encoder layers process them, and finally, the classification layer maps the final hidden state to logits. We refer to this standard formulation as token embedding.

Approach 1: extending vocabulary. The standard token embedding is used. Introducing the auxiliary modulus Kq needs two minor modifications to the standard training pipeline. First, the vocabulary set is extended from $\{0, \ldots, q-1\}$ to $\{0, \ldots, Kq-1\}$. Second, one of the two labels $f_q(\mathbf{x})$ and $f_{Kq}(\mathbf{x})$ with probability r and $1-r$, respectively, is selected for the classification loss (i.e., the cross-entropy loss), as shown in Equation (2). In inference, only the first q logits are used to predict the solution to modular addition with modulus q.

While this approach is simple, it can suffer from a large vocabulary size when K is large. To address this, we extend the angular embedding approach [26] to handle the auxiliary modulus Kq.

Approach 2: dual angular embedding. Angular embedding [26] embeds a scalar $x \in \{0, \ldots, q-1\}$ to a two-dimensional unit circle $(\cos\phi, \sin\phi)$ with $\phi = 2\pi x_i/q$ and then lifts it to the embedding space by a linear layer. The linear read-out layer maps the final hidden state to a 2-dimensional vector. The MSE loss is used to measure the angular error, instead of the classification loss. In our case, we use two moduli q and Kq, and thus scalars are embedded to a four-dimensional unit circle $(\cos\phi, \sin\phi, \cos\phi', \sin\phi')$ with $\phi' = 2\pi x_i/Kq$. The read-out layer maps the final hidden state to a 4-dimensional vector. In conclusion, only the first two dimensions are used to predict the solution to modular addition with modulus q.

5 Experiment

In this section, we evaluate the proposed method on the modular addition task across various numbers of terms N and moduli q. Particularly, we show that our method attains comparable or higher match accuracy than the sparse method [24] using only one-tenth of the training samples (Section 5.1) for most of the settings.

5.1 Exact modular addition

First, we evaluate the proposed method on the exact modular addition task, where models are trained to predict the exact solution to modular addition. Most experimental setups follow [24] unless otherwise emphasized.

Dataset. For the training dataset $\mathcal{D}_{\text{train}}$ parameterized by N, q, and $|\mathcal{D}_{\text{train}}|$, each entry of the input $\mathbf{x}$ was sampled from a uniform distribution for our method and from the sparse distribution for the sparse method [24]. The test dataset $\mathcal{D}_{\text{test}}$ was fixed to 1M uniformly generated samples. For each input $\mathbf{x}$, the dataset provided the primary label $f_q(\mathbf{x})$ and the quotient $c = \lfloor \sum_{i=1}^{N} x_i/q \rfloor$. Since our method required the auxiliary target $f_{Kq}(\mathbf{x})$ during training, it was dynamically generated by first reconstructing the total sum as $c \cdot q + f_q(\mathbf{x})$ and then applying the modulo Kq operation.

Training setup. An encoder-only Transformer architecture consisting of 4 layers and 4 attention heads, with an embedding dimension of 256 and a feed-forward network (FFN) dimension of 2048 was adopted. Either token embedding (cf. **Approach 1**) or angular embedding (cf. **Approach 2**) was used, as detailed in Section 4.3. A custom loss to avoid embedding collapse was also used for the sparse method as Saxena et al. [24] suggested. The dropout rate was set to 0.0, and positional embeddings were not used. For optimization, the AdamW optimizer [18] was used with $(\beta_1, \beta_2) = (0.9, 0.999)$ and a weight decay of 0.1. All models were trained for 10 epochs with a batch size of 250. The learning rate was set to 3×10^{-5} and scheduled with a linear decay, using a warmup ratio of 0.05 rather than fixed steps to accommodate varying training sample sizes. All experiments were conducted on a single GPU of NVIDIA RTX 6000 Ada. Training time per model ranged from approximately 1 minute to 20 hours, depending on N, q, and the training sample size.

Hyperparameter search. The optimal hyperparameter pair $(K, r) \in \{4, \ldots, 9\} \times \{0.1, \ldots, 0.4\}$ was determined via a grid search on a small dataset ($|\mathcal{D}_{\text{train}}| = 100$K) and reused at every sample size. The specific values of K and r for each setting are detailed in Section D. This search only required observing relative performance differences rather than full convergence. For the sparse baseline we did not search over sparsity schedules, and instead adopted $g(z) \propto 1/\sqrt{N-z+1}$, the choice reported by Saxena et al. [24]. Section 5.3 shows that the method is robust to (K, r): even the lowest accuracy in the grid of Table 3 exceeds the sparse baseline.

Table 1: Comparison of match accuracy (%) for **(a)** token embedding and **(b)** angular embedding. We report our method, the sparse baseline [24], and Uniform (uniform inputs, modulus q only) at both 100K and 1M. The sparse baseline is adapted to the embedding type as detailed in the main text.

(a) Token embedding

		100K			1M		
N	q	Ours	Sparse	Uniform	Ours	Sparse	Uniform
8	31	**89.1**	3.2	3.0	**99.4**	98.6	2.8
	97	**13.9**	0.9	0.9	**90.1**	74.9	1.0
	257	**1.0**	0.5	0.4	**34.0**	0.5	0.3
16	31	**58.4**	3.3	3.0	**98.9**	80.7	3.5
	97	**6.2**	1.0	0.9	**81.9**	1.1	1.1
	257	**0.6**	0.4	0.4	**21.7**	0.4	0.4
32	31	**33.1**	2.9	3.1	**96.5**	38.7	3.0
	97	**1.7**	1.1	0.9	**62.2**	1.0	1.1
	257	**0.5**	0.4	0.5	**12.6**	0.3	0.3

(b) Angular embedding

		100K			1M		
N	q	Ours	Sparse	Uniform	Ours	Sparse	Uniform
16	97	**98.9**	32.2	27.5	**99.9**	92.6	95.9
	257	**93.0**	11.8	6.4	**99.4**	63.8	72.2
	433	**80.7**	6.8	8.9	**98.5**	45.7	50.2
32	97	**95.6**	4.2	0.9	**99.6**	81.1	1.2
	257	**73.5**	0.6	0.4	**88.5**	44.9	0.4
	433	**57.5**	0.9	0.3	**92.2**	27.7	0.2
64	97	**83.2**	0.9	1.1	**97.4**	58.7	1.0
	257	**43.2**	0.3	0.3	**81.1**	23.4	0.4
	433	**29.0**	0.2	0.3	**53.8**	11.4	0.2

Evaluation. Performance was measured by match accuracy on $\mathcal{D}_{\text{test}}$, defined as the proportion of predictions that perfectly matched the ground-truth labels. In the case of angular embedding, the model outputs a point $(\cos\hat{\phi}, \sin\hat{\phi})$, which was first converted back into an angle $\hat{\phi} \in [0, 2\pi)$. This angle was then mapped to a continuous value $\hat{s} = \hat{\phi}q/(2\pi) \in [0, q)$, and finally rounded to the nearest integer before evaluating the match.

Results: scalability and sample efficiency. As shown in Table 1, our method consistently outperforms the baseline across all configurations. For token embeddings (Table 1a), the baseline completely fails to learn with 100K samples, whereas our method successfully initiates learning (e.g., 89.1% at $N = 8, q = 31$). Our approach also exhibits superior scalability: when increasing N from 8 to 32 (using 1M samples, $q = 31$), the baseline's accuracy collapses from 98.6% to 38.7%, while ours degrades minimally ($99.4\% \rightarrow 96.5\%$). Under angular embeddings (Table 1b), our method demonstrates even extreme sample efficiency, achieving higher accuracies with just 100K samples than the baseline does with 1M. For instance, at $N = 16, q = 257$, our model reaches 93.0% (100K) compared to the baseline's 63.8% (1M), with similar trends observed even at larger scales (e.g., $N = 64, q = 97$). Uniform in Table 1 uses uniform inputs and modulus q only. Refer Sections E and F for repeated runs and staged training under the same task.

5.2 Larger-scale modular addition

We next address larger-scale modular addition tasks: two integers $N \in \{16, 32, 64, 128\}$ and $q \in \{257, 3329, 42899, 974269\}$. This was particularly targeted by the sparse method in the context of attack on LWE. In this case, the evaluation metric (τ-accuracy with $\tau \in \{0.005, 0.01\}$) was used.

$$\tau\text{-accuracy} = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{x \in \mathcal{D}_{\text{test}}} \mathbb{1}_{\|\hat{y}-y\| \leq \tau q}$$

where $\hat{y}$ is the continuous value decoded from the model's output for the input $\mathbf{x}$, and y is its target label. The distance $\|\hat{y} - y\|$ is $\min(|\hat{y} - y|, q - |\hat{y} - y|)$, so residues modulo q wrap around (i.e., 0 and $q - 1$ are adjacent). The indicator $\mathbb{1}_S$ equals 1 when its argument lies in S and 0 otherwise. This metric relaxes exact match by scoring predictions that lie within a margin τq of the target.

Results. The proposed method maintains high efficiency for sample size and scalability even in large-scale modular addition. As shown in Table 2, it attains higher τ-accuracy than the baseline at the same sample size. At $N = 128$ with 1M samples, the baseline hardly improves. Our method remains high. With 1M samples, our method matches the baseline trained on 10M samples. At $N = 128$, it surpasses that result. Section G shows how τ-accuracy changes as τ varies for $N = 128$, $q = 974269$, and the α-worst-case τ-bound which is defined as the minimum τ needed to achieve $\alpha\%$ τ-accuracy.

Table 2: Comparison of τ-accuracy (%) ($\tau \in \{0.005, 0.01\}$) between our method and the baseline [24] using 1M and 10M training samples for angular embedding. Our method consistently achieves higher τ-accuracy even in the large-scale modular addition settings based on the prior work.

		$\tau = 0.01$ Accuracy (%)				$\tau = 0.005$ Accuracy (%)			
		1M		10M		1M		10M	
N	q	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline
16	257	**100.0**	98.4	**100.0**	100.0	**100.0**	92.1	**100.0**	99.8
	3329	**100.0**	98.3	**100.0**	100.0	**100.0**	92.5	**100.0**	99.9
	42899	**100.0**	98.6	**100.0**	100.0	**99.9**	93.3	**100.0**	99.9
	974269	**100.0**	98.4	**100.0**	100.0	**100.0**	93.2	**100.0**	99.9
32	257	**100.0**	95.9	**100.0**	99.9	**99.1**	82.4	**100.0**	99.6
	3329	**99.9**	95.9	**100.0**	99.9	**99.2**	82.3	**100.0**	99.5
	42899	**100.0**	95.6	**100.0**	99.9	**99.4**	81.5	**100.0**	99.5
	974269	**99.7**	95.2	**100.0**	100.0	**97.8**	79.9	**100.0**	99.5
64	257	**99.9**	80.2	**100.0**	99.8	**98.1**	53.3	**100.0**	98.8
	3329	**99.8**	70.5	**100.0**	99.8	**98.0**	43.4	**100.0**	98.7
	42899	**99.7**	62.1	**100.0**	99.7	**97.6**	36.5	**100.0**	98.2
	974269	**99.7**	82.4	**100.0**	99.7	**97.8**	55.8	**100.0**	98.5
128	257	**99.1**	10.3	**100.0**	99.1	**96.4**	5.0	**100.0**	93.9
	3329	**99.6**	7.6	**100.0**	99.0	**97.2**	3.9	**100.0**	93.1
	42899	**98.4**	23.4	**100.0**	99.1	**94.4**	11.9	**99.6**	93.8
	974269	**99.4**	18.6	**100.0**	99.2	**97.0**	9.5	**100.0**	93.9

5.3 Sensitivity to hyperparameters

We evaluate the hyperparameter sensitivity of our approach by reporting the average, minimum, and maximum match accuracies over the grid search space ($K \in \{4, \ldots, 9\}, r \in \{0.1, \ldots, 0.4\}$). These results are then compared against the baseline from Section 5.1.

Robustness to hyperparameter selection. As summarized in Table 3, our method demonstrates highly stable learning performance across various combinations of K and r. The narrow gap between the minimum and maximum accuracies indicates that the model yields consistently high performance without relying on heavy parameter tuning. Furthermore, even the minimum accuracy within this search space shows a clear improvement over the baseline result from Section 5.1. For instance, under the $N = 64, q = 433$ setting with angular embedding, our minimum accuracy of 34.0% exceeds the baseline's 11.4%. Similarly, for token embedding ($N = 16, q = 97$), the worst-case accuracy in our grid still substantially surpasses the baseline's 1.1%. This confirms that our approach is inherently robust to hyperparameter choices. Detailed results across a broader range ($K \in \{2, \ldots, 10\}, r \in \{0.1, \ldots, 0.9\}$) are provided in Section H.

Relationship between accuracy and difficulty. While the proposed method is robust on average, the accuracy can fluctuate significantly depending on the configuration. Figure 3 presents a heatmap of a setting with such high variance to examine the underlying trends. The orange box highlights the region where the ratio ρ in Equation (4) lies between 0.8 and 1.2. As shown in Section 4.2, $\rho = 1$ equates the expected number of wraps of the proposed method with that of the sparse method. High accuracy is neither confined to this box nor guaranteed within it, yet the strongest settings, such as $K \in \{5, \ldots, 7\}$ and $r \in \{0.4, 0.5\}$, lie near $\rho = 1$.

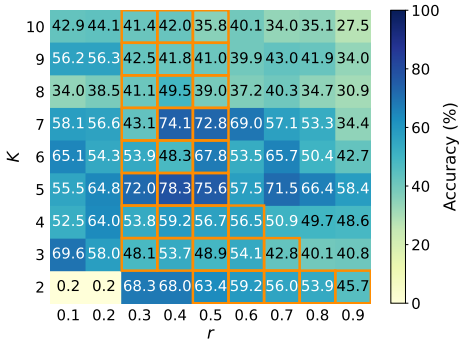


Figure 3: Match accuracy for angular embeddings ($N = 64$, $q = 433$). The orange box marks $0.8 \leq \rho \leq 1.2$ in Equation (4).

Table 3: Minimum, average, and maximum match accuracy (%) of the proposed method calculated across the practical grid search space ($K \in \{4, \ldots, 9\}, r \in \{0.1, \ldots, 0.4\}$). For reference, the baseline result from Table 1 is also provided. The improvement column reports average accuracy minus baseline (%). Our method achieves higher match accuracy even on average, demonstrating strong hyperparameter robustness.

Embedding	Problem Setting		Ours	Baseline	Gain
	N	q	Min /Avg. /Max	(from Table 1)	Avg. − Base.
Token	8	31	99.2 / **99.5** / 99.6	98.6	+0.90
		97	90.1 / **93.1** / 96.7	74.9	+18.2
		257	27.3 / **33.8** / 37.0	0.5	+33.3
	16	31	98.1 / **98.9** / 99.4	80.7	+18.2
		97	76.1 / **81.1** / 84.2	1.1	+80.0
		257	14.2 / **23.9** / 27.1	0.4	+23.5
	32	31	92.7 / **97.2** / 98.5	38.7	+58.5
		97	46.0 / **63.6** / 69.6	1.0	+62.6
		257	0.4 / **13.5** / 17.8	0.3	+13.2
Angular	16	97	99.7 / **99.9** / 100	92.6	+7.30
		257	97.4 / **99.1** / 99.8	63.8	+35.3
		433	90.2 / **96.6** / 99.4	45.7	+50.9
	32	97	98.0 / **99.3** / 99.9	81.1	+18.2
		257	84.3 / **93.5** / 99.0	44.9	+48.6
		433	61.8 / **83.8** / 94.4	27.7	+56.1
	64	97	91.8 / **96.5** / 99.5	58.7	+37.8
		257	64.7 / **80.5** / 94.1	23.4	+57.1
		433	34.0 / **54.7** / 78.4	11.4	+43.3

6 Conclusion

We proposed a novel training method for stable modular addition learning by introducing an expanded modulus Kq as an auxiliary task alongside the target modulus q. Unlike the sparse method [24], which changes the input distribution during training, our approach keeps the input distribution unchanged and thus avoids this source of covariate shift. Theoretical analysis showed that our method reduces problem difficulty similarly to sparse training, but without modifying the input distribution.

Experimental results demonstrated that our method improves learning efficiency and accuracy across various conditions of terms N, moduli q, and data sizes. Notably, our method achieves high accuracy even under conditions of large N, large q, and small data sizes, where baseline methods fail to learn. Furthermore, it achieves equal or superior match accuracy and τ-accuracy even when the training data size is reduced to one-tenth. These results indicate that methods avoiding distribution shifts possess high learning efficiency and scalability.

Limitations and future work. Empirical results in this paper use encoder-only Transformers. The method changes only the training objective, and other architectures, including MLPs, remain untested. The contribution is limited to modular addition. A preliminary check of whether the auxiliary-task idea extends to modular multiplication is reported in Section I; scaling that extension remains future work. Establishing a more efficient optimization method for the hyperparameters K and r, which are currently determined by grid search, remains an important challenge. Investigating the relationship between the theoretical difficulty ratio in Equation (4) and empirical performance is left for future work. Based on this relationship, we aim to develop a principled search strategy that constrains the parameter space. We will also conduct a detailed analysis of how the problem difficulty changes.

Acknowledgments

This research was partially supported by JST PRESTO Grant Number JPMJPR24K4, JST BOOST Program Grant Number JPMJBY24C6, and JSPS Program for Forming Japan's Peak Research Universities (J-PEAKS) Grant Number JPJS00420230002, Mitsubishi Electric Information Technology R&D Center, and JSPS KAKENHI Grant Number 26K02996.

References

[1] Emmanuel Abbe, Elisabetta Cornacchia, and Aryo Lotfi. Provable advantage of curriculum learning on parity targets with mixed inputs. In *Advances in Neural Information Processing Systems*, volume 36, pages 24291–24321. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/4c8ce3c63f6b66d6811c6d67c68e487b-Paper-Conference.pdf.

[2] Alberto Alfarano, François Charton, and Amaury Hayat. Global lyapunov functions: a long-standing open problem in mathematics, with symbolic transformers. In *Advances in Neural Information Processing Systems*, volume 37, pages 93643–93670, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/aa280e73c4e23e765fde232571116d3b-Paper-Conference.pdf.

[3] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, page 41–48, 2009. URL https://doi.org/10.1145/1553374.1553380.

[4] Csaba Both, Benjamin Hoover, Hendrik Strobelt, Dmitry Krotov, Daniel Karl I. Weidele, Mauro Martino, and Nima Dehmamy. Small models, smarter learning: The power of joint task training, 2026. URL https://arxiv.org/abs/2505.18369.

[5] Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

[6] Francois Charton, Amaury Hayat, and Guillaume Lample. Learning advanced mathematical computations from examples. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=-gfhS00XfKj.

[7] Amit Daniely and Eran Malach. Learning parities with neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 20356–20365, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/eaae5e04a259d09af85c108fe4d7dd0c-Paper.pdf.

[8] Darshil Doshi, Tianyu He, Aritra Das, and Andrey Gromov. Grokking modular polynomials, 2024. URL https://arxiv.org/abs/2406.03495.

[9] Hiroki Furuta, Gouki Minegishi, Yusuke Iwasawa, and Yutaka Matsuo. Towards empirical interpretation of internal circuits and properties in grokked transformers on modular polynomials. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=MzSf70uXJO.

[10] Andrey Gromov. Grokking modular arithmetic, 2023. URL https://arxiv.org/abs/2301.02679.

[11] Michael Hahn and Mark Rofin. Why are sensitive functions hard for transformers? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14973–15008, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.800. URL https://aclanthology.org/2024.acl-long.800/.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, page 1026–1034, 2015. URL https://doi.org/10.1109/ICCV.2015.123.

[13] Hiroshi Kera, Yuki Ishihara, Yuta Kambe, Tristan Vaccon, and Kazuhiro Yokoyama. Learning to compute gröbner bases. In *Advances in Neural Information Processing Systems*, volume 37, pages 33141–33187, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/3a1de90699eec7d7f42c91d81f94af16-Paper-Conference.pdf.

[14] Hiroshi Kera, Nico Pelleriti, Yuki Ishihara, Max Zimmer, and Sebastian Pokutta. Computational algebra with attention: Transformer oracles for border basis algorithms. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=bA9rhgW0Hk.

[15] Juno Kim and Taiji Suzuki. Transformers provably solve parity efficiently with chain of thought. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=n2NidsYDop.

[16] 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.

[17] Ziming Liu, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 34651–34663, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/dfc310e81992d2e4cedc09ac47eff13e-Paper-Conference.pdf.

[18] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[19] Gavin McCracken, Gabriela Moisescu-Pareja, Vincent Létourneau, Doina Precup, and Jonathan Love. Uncovering a universal abstract algorithm for modular addition in neural networks. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=zuHs6RHQwT.

[20] Mohamad Amin Mohamadi, Zhiyuan Li, Lei Wu, and Danica J. Sutherland. Why do you grok? a theoretical analysis on grokking modular addition. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=ad5I6No9G1.

[21] 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.

[22] Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets, 2022. URL https://arxiv.org/abs/2201.02177.

[23] Sebastian Ruder. An overview of multi-task learning in deep neural networks, 2017. URL https://arxiv.org/abs/1706.05098.

[24] Eshika Saxena, Alberto Alfarano, Emily Wenger, and Kristin E. Lauter. Making hard problems easier with custom data distributions and loss regularization: A case study in modular arithmetic. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=le8hVvWi6Q.

[25] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. Failures of gradient-based deep learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3067–3075, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/shalev-shwartz17a.html.

[26] Samuel Stevens, Emily Wenger, Cathy Yuanchen Li, Niklas Nolte, Eshika Saxena, Francois Charton, and Kristin E. Lauter. Salsa fresca: Angular embeddings and pre-training for ML attacks on learning with errors. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=w4nd5695sq.

[27] NIST US Department of Commerce. Nist releases first 3 finalized post-quantum encryption standards, 2024. https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards.

[28] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

[29] Noam Wies, Yoav Levine, and Amnon Shashua. Sub-task decomposition enables learning in sequence to sequence tasks. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=BrJATVZDWEH.

[30] 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, pages 27223–27250, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/56cbfbf49937a0873d451343ddc8c57d-Paper-Conference.pdf.

A Formal restatement and proof of Proposition 1

Proposition 1 in the main text is informal in two respects: it leaves the loss unspecified and identifies "the expected loss on a uniformly random zero-free input" without fixing a measure. The formal restatement below makes both precise; Table 4 reports the numerical value of Equation (1) for parameter ranges typical of modular-addition experiments.

Proposition 3 (Formal version of Proposition 1)**.** *Let $\mathcal{X} := \{0, \dots, q-1\}^N$ and $n_0(x) := \#\{i \in [N] : x_i = 0\}$. Suppose P_{te} is uniform on $\mathcal{X}$, and P_{tr} is the sparse construction of Saxena et al. [24]: sample $z \sim g$ with $g(z) \propto 1/\sqrt{N-z+1}$ for $z \in \{1, \dots, N\}$, choose z positions uniformly at random, fill them independently and uniformly from $\{0, \dots, q-1\}$, and set the remaining $N-z$ positions to zero. Let $L_f(x) \geq 0$ be a non-negative pointwise loss, and let $\bar{L}_f(0) := \mathbb{E}_{x \sim \mathrm{Unif}(\{x : n_0(x)=0\})}[L_f(x)]$ be the loss averaged over a uniformly random zero-free input. If $\bar{L}_f(0) \geq \varepsilon$ and $R_{\mathrm{tr}}(f) \leq \delta$, then*

$$R_{\mathrm{te}}(f) - R_{\mathrm{tr}}(f) \;\geq\; \varepsilon\Big(1 - \tfrac{1}{q}\Big)^N - \delta.$$

Proof. P_{te} is permutation-invariant in the coordinates of x, so conditional on $n_0(x) = 0$ it is uniform on the set $\{x : n_0(x) = 0\}$ of zero-free inputs. Therefore $\mathbb{E}_{P_{\mathrm{te}}}[L_f(x) \mid n_0(x) = 0] = \bar{L}_f(0)$. Decomposing $R_{\mathrm{te}}(f)$ by conditioning on $n_0(x)$ and dropping the non-negative contributions from $n_0(x) \neq 0$,

$$R_{\mathrm{te}}(f) \;=\; \sum_{k=0}^{N} P_{\mathrm{te}}(n_0(x) = k)\, \mathbb{E}_{P_{\mathrm{te}}}[L_f(x) \mid n_0(x) = k] \;\geq\; P_{\mathrm{te}}(n_0(x) = 0)\, \bar{L}_f(0) \;\geq\; \Big(1 - \tfrac{1}{q}\Big)^N \varepsilon,$$

where the last step uses $P_{\mathrm{te}}(n_0(x) = 0) = \Pr_{Z \sim \mathrm{Bin}(N, 1/q)}[Z = 0] = (1 - 1/q)^N$ (since under the uniform distribution each entry independently equals 0 with probability $1/q$) and $\bar{L}_f(0) \geq \varepsilon$. Subtracting $R_{\mathrm{tr}}(f) \leq \delta$ yields the bound. □

Table 4: Numerical lower bound $\mathrm{LB}(q, N) := (1 - 1/q)^N$ (the prefactor in Equation (1)) for parameter ranges typical of modular-addition experiments. The reported value bounds $R_{\mathrm{te}}(f) - R_{\mathrm{tr}}(f)$ from below by $\varepsilon \cdot \mathrm{LB}(q, N) - \delta$.

$N \backslash q$	17	23	47	113
8	0.61	0.70	0.84	0.93
16	0.38	0.49	0.71	0.87
24	0.23	0.34	0.60	0.81
32	0.14	0.24	0.50	0.75

B Robustness to Model Architecture Variations

We experimentally observed that the performance of the sparse method in Saxena et al. [24] is highly sensitive to the underlying model architecture. To systematically investigate the extent of this instability, we evaluated the match accuracy across 16 different model configurations. Our results demonstrate that the baseline method suffers from severe performance fluctuations, heavily depending on specific architectural choices.

Following the base experimental setup of Section 5.1. These 16 configurations were generated by exhaustively combining four key architectural hyperparameters: the position of the normalization layer, the inclusion of bias terms, the weight initialization strategy, and the dropout rate. The specific settings for each component are detailed in Table 5. The experiment was conducted using token embedding with a sequence length of $N = 8$ and a modulus of $q = 97$.

Table 6 presents the match accuracy across all 16 configurations. The results clearly illustrate that the sparse method in Saxena et al. [24] is highly sensitive to minor architectural choices, frequently suffering from severe performance degradation. For example, under the Pre-Norm and Default initialization setting without dropout, simply removing the bias terms causes the baseline's accuracy

to plummet drastically from 74.9% to a mere 1.2%. Such extreme fragility to standard model variations serves as a strong motivation for exploring a new, more robust approach. While our proposed method also exhibits some performance variations depending on the configuration (such as a general decrease in accuracy when dropout is applied), the magnitude of these drops is significantly smaller. In the same example of removing the bias terms, our method maintains a consistently high accuracy (shifting only from 90.1% to 91.5%), thereby confirming its superior structural stability.

Table 5: Details of the hyperparameter choices for the model architecture variations: normalization layer, bias term, weight initialization, and dropout.

Component	Value	Description
Normalization	Pre-Norm	Layer normalization is applied before each sub-layer.
	Post-Norm	Layer normalization is applied after each sub-layer.
Bias Term	Incl.	Bias terms are included in all learnable layers.
	Excl.	Bias terms are removed from all learnable layers.
Weight Initialization	$\sigma = 0.02$	Linear and embedding layers are initialized from a normal distribution $\mathcal{N}(0, 0.02^2)$.
	Default	Embedding layers are initialized from $\mathcal{N}(0, 1)$, and linear layers use Kaiming initialization [12].
Dropout	0.0	No dropout is applied.
	0.1	Dropout rate is set to 0.1.

Table 6: Match accuracy (%) across 16 model configurations for token embedding ($N = 8, q = 97$). The sparse method in Saxena et al. [24] suffers from severe instability, whereas the proposed method consistently achieves high match accuracy regardless of the architecture.

Model Configuration				Match Accuracy (%)	
Normalization	Bias	Initialization	Dropout	Sparse[24]	Ours
Pre-Norm	Incl.	$\sigma = 0.02$	0.0	5.1	73.5
			0.1	10.2	33.9
		Default	0.0	74.9	90.1
			0.1	97.2	17.5
	Excl.	$\sigma = 0.02$	0.0	1.5	77.7
			0.1	4.8	33.8
		Default	0.0	1.2	91.5
			0.1	98.5	14.2
Post-Norm	Incl.	$\sigma = 0.02$	0.0	5.7	75.8
			0.1	25.3	35.2
		Default	0.0	6.2	92.5
			0.1	18.6	33.3
	Excl.	$\sigma = 0.02$	0.0	1.1	74.0
			0.1	0.9	35.0
		Default	0.0	7.2	90.1
			0.1	23.8	27.4

C A proof of Proposition 2 and more analysis of Equation (3)

In this section, we give a proof of Proposition 2 , and describe properties of a random variable in Equation (3), which represents the number of wraps around the modulus in the proposed methods. We introduce the notations used throughout this section. Let N, K and q be positive integers. Let

$\mathbf{x} = [x_1, \ldots, x_N]$ be a vector of N i.i.d. random variables with a discrete distribution $U_q(x) = 1/q$ if $x \in \{0, \ldots, q-1\}$ otherwise 0. Put $S_M = \sum_{i=1}^{M} x_i$ for $0 \leq M \leq N$.

C.1 A proof of Proposition 2

Let z be a random variable that represents the number of non-zero entries of an input data, and has a distribution $f_{\text{inv_sqrt}}(z) \propto 1/\sqrt{N-z+1}$ introduced in Saxena et al. [24]; explicitly, $g(z) = 1/(C_N\sqrt{N-z+1})$ where $C_N = \sum_{z=0}^{N} 1/\sqrt{N-z+1}$. We define three random variables Y_0, Y_1 and Y_2 as follows:

$$Y_0 = \frac{S_N}{q}, \qquad Y_1 = (1-r)\frac{S_N}{q} + r\frac{S_N}{Kq}, \qquad Y_2 = \frac{S_z}{q}.$$

Lemma 1. *Keep the above notation. For Y_0, Y_1 and Y_2, we have*

$$\frac{\mathbb{E}[Y_0]}{N} = \frac{q-1}{2q}, \qquad \frac{\mathbb{E}[Y_1]}{N} = \left((1-r) + r\frac{1}{K}\right)\frac{q-1}{2q}, \qquad \lim_{N\to\infty}\frac{\mathbb{E}[Y_2]}{N} = \frac{2}{3}\,\frac{q-1}{2q}.$$

Proof. Since x_i's are i.i.d. satisfying $\mathbb{E}[x_i] = \sum_{j=0}^{q-1} j/q = (q-1)/2$, we have $\mathbb{E}[S_N] = N(q-1)/2$. Hence, the expectations of Y_0, Y_1 and Y_2 are given by

$$\begin{aligned}
\mathbb{E}[Y_0] &= \mathbb{E}\left[\frac{S_N}{q}\right] = \frac{N(q-1)}{2q},\\
\mathbb{E}[Y_1] &= \mathbb{E}\left[(1-r)\frac{S_N}{q} + r\frac{S_N}{Kq}\right] = \left((1-r) + r\frac{1}{K}\right)\frac{N(q-1)}{2q},\\
\mathbb{E}[Y_2] &= \sum_{z=0}^{N}\frac{1}{C_N}\frac{1}{\sqrt{N-z+1}}\mathbb{E}\left[\frac{S_z}{q}\right] = \sum_{z=0}^{N}\frac{1}{C_N}\frac{1}{\sqrt{N-z+1}}\frac{z(q-1)}{2q}\\
&= \frac{q-1}{2q}\sum_{z=0}^{N}\frac{z}{C_N\sqrt{N-z+1}} = \frac{q-1}{2q}\frac{\sum_{z=0}^{N}\frac{z}{\sqrt{N-z+1}}}{\sum_{z=0}^{N}\frac{1}{\sqrt{N-z+1}}}.
\end{aligned}$$

Hence, the statement holds for Y_0 and Y_1. We prove the statement for Y_2. The following hold:

$$\begin{aligned}
\sum_{z=0}^{N}\frac{z}{\sqrt{N-z+1}} &= \sum_{k=1}^{N+1}\frac{N-k+1}{\sqrt{k}} = (N+1)\sum_{k=1}^{N+1}\frac{1}{\sqrt{k}} - \sum_{k=1}^{N+1}\sqrt{k}\\
&= (N+1)\sqrt{N}\sum_{k=1}^{N+1}\frac{1}{\sqrt{k/N}}\frac{1}{N} - N\sqrt{N}\sum_{k=1}^{N+1}\sqrt{k/N}\frac{1}{N},
\end{aligned}$$

and

$$\sum_{z=0}^{N}\frac{1}{\sqrt{N-z+1}} = \sum_{k=1}^{N+1}\frac{1}{\sqrt{k}} = \sqrt{N}\sum_{k=1}^{N+1}\frac{1}{\sqrt{k/N}}\frac{1}{N}.$$

Hence, we have

$$\begin{aligned}
\frac{\mathbb{E}[Y_2]}{N} &= \frac{1}{N}\frac{q-1}{2q}\frac{\sum_{z=0}^{N}\frac{z}{\sqrt{N-z+1}}}{\sum_{z=0}^{N}\frac{1}{\sqrt{N-z+1}}}\\
&= \frac{q-1}{2q}\frac{(N+1)\sqrt{N}\sum_{k=1}^{N+1}\frac{1}{\sqrt{k/N}}\frac{1}{N} - N\sqrt{N}\sum_{k=1}^{N+1}\sqrt{k/N}\frac{1}{N}}{N\sqrt{N}\sum_{k=1}^{N+1}\frac{1}{\sqrt{k/N}}\frac{1}{N}}\\
&= \frac{q-1}{2q}\frac{(1+1/N)\sum_{k=1}^{N+1}\frac{1}{\sqrt{k/N}}\frac{1}{N} - \sum_{k=1}^{N+1}\sqrt{k/N}\frac{1}{N}}{\sum_{k=1}^{N+1}\frac{1}{\sqrt{k/N}}\frac{1}{N}}
\end{aligned}$$

Now, we compute the limits of the numerators and the denominators as follows:

$$\lim_{N\to\infty}\sum_{k=1}^{N+1}\frac{1}{\sqrt{k/N}}\frac{1}{N}=\int_0^1\frac{1}{\sqrt{x}}dx=2$$
$$\lim_{N\to\infty}\sum_{k=1}^{N+1}\sqrt{k/N}\frac{1}{N}=\int_0^1\sqrt{x}dx=\frac{2}{3}.$$

Therefore, the desired limit is given by

$$\lim_{N\to\infty}\frac{\mathbb{E}[Y_2]}{N}=\frac{q-1}{2q}\frac{2-\frac{2}{3}}{2}=\frac{2}{3}\frac{q-1}{2q}.$$

□

Lemma 2. *Let Y_0, Y_1 and Y_2 be as in Lemma 1, and let X_0, X_1 and X_2 be as in Section 4. Then for $i=0,1,2$, we have*

$$0\le\mathbb{E}[Y_i]-\mathbb{E}[X_i]\le 1,$$

Proof. For every non-negative real number t, $0\le t-\lfloor t\rfloor<1$. Therefore $0\le Y_0-X_0<1$ and $0\le Y_2-X_2<1$. Moreover, we have

$$Y_1-X_1=(1-r)\left(\frac{S_N}{q}-\left\lfloor\frac{S_N}{q}\right\rfloor\right)+r\left(\frac{S_N}{Kq}-\left\lfloor\frac{S_N}{Kq}\right\rfloor\right),$$

and the two coefficients are non-negative and sum to 1, so $0\le Y_1-X_1<1$. □

Proposition 4 (Proposition 2)**.** *Keep the above notation. For Y_0, we have*

$$\frac{\mathbb{E}[Y_0]}{N}=\frac{q-1}{2q}.$$

Fixed q, K and r, we have

$$\lim_{N\to\infty}\frac{\mathbb{E}[X_0]}{N}=\frac{q-1}{2q},\quad\lim_{N\to\infty}\frac{\mathbb{E}[X_1]}{N}=\left((1-r)+r\frac{1}{K}\right)\frac{q-1}{2q},\quad\lim_{N\to\infty}\frac{\mathbb{E}[X_2]}{N}=\frac{2}{3}\frac{q-1}{2q}.$$

Proof. The identity $\mathbb{E}[Y_0]/N=(q-1)/(2q)$ is the first claim of Lemma 1. By Lemma 2, for $i=0,1,2$, we have

$$0\le\frac{\mathbb{E}[Y_i]}{N}-\frac{\mathbb{E}[X_i]}{N}\le\frac{1}{N},$$

Hence, by Lemma 1 and the above inequalities, we have

$$\lim_{N\to\infty}\frac{\mathbb{E}[X_0]}{N}=\lim_{N\to\infty}\frac{\mathbb{E}[Y_0]}{N}=\frac{q-1}{2q},$$
$$\lim_{N\to\infty}\frac{\mathbb{E}[X_1]}{N}=\lim_{N\to\infty}\frac{\mathbb{E}[Y_1]}{N}=\left((1-r)+r\frac{1}{K}\right)\frac{q-1}{2q},$$
$$\lim_{N\to\infty}\frac{\mathbb{E}[X_2]}{N}=\lim_{N\to\infty}\frac{\mathbb{E}[Y_2]}{N}=\frac{2}{3}\frac{q-1}{2q}.$$

□

C.2 Properties of a random variable in Equation (3)

We describe properties of a random variable, $D_{Kq}(\mathbf{x})$, which represents the number of wraps around the modulus. We recall the definition.

$$D_{Kq}(\mathbf{x})=\left\lfloor\frac{S_N}{Kq}\right\rfloor,\tag{5}$$

We begin with the following proposition on the distribution of S_N:

Proposition 5. *Keep the above notation. The distribution of S_N is defined by*

$$U_q^{*N}(S_N = s) = \frac{1}{q^N} \sum_{k=0}^{\lfloor \frac{s}{q} \rfloor} (-1)^k \binom{N}{k} \binom{s - qk + N - 1}{N - 1},$$

where U_q^{*N} is the convolution of N copies U_q.

Proof. Let $G_q(z)$ be the probability generating function of x_i's and let $H_{q,N}(z)$ be the probability generating function of S_N. Since x_i's are i.i.d. and $G_q(z) = \sum_{k=0}^{q-1} z^k/q$, for $z \in (-1, 1)$ we have

$$\begin{aligned} H_{q,N}(z) &= G_q(z)^N \\ &= \frac{1}{q^N} \left(\sum_{k=0}^{q-1} z^k \right)^N \\ &= \frac{1}{q^N} \left(\frac{1 - z^q}{1 - z} \right)^N \end{aligned}$$

On the open interval $(-1, 1)$, $1/(1 - z)$ is equal to the power series $\sum_{k=0}^{\infty} z^k$ because its radius of convergence is 1. Therefore, we have

$$\begin{aligned} H_{q,N}(z) &= \frac{1}{q^N} \left(\frac{1 - z^q}{1 - z} \right)^N \\ &= \frac{1}{q^N} \left((1 - z^q) \sum_{k=0}^{\infty} z^k \right)^N \\ &= \frac{1}{q^N} (1 - z^q)^N \left(\sum_{k=0}^{\infty} z^k \right)^N \\ &= \frac{1}{q^N} \left(\sum_{l=0}^{N} (-1)^l \binom{N}{l} z^{lq} \right) \left(\sum_{k=0}^{\infty} \binom{k + N - 1}{N - 1} z^k \right). \end{aligned}$$

Hence, the coefficient of z^s ($s \geq 0$) in $H_{q,N}(z)$ is given by

$$\begin{aligned} \frac{1}{q^N} \sum_{\substack{k,l \geq 0 \\ lq + k = s}}^{N} (-1)^l \binom{N}{l} \binom{k + N - 1}{N - 1} &= \frac{1}{q^N} \sum_{\substack{\frac{s}{q} \geq l \geq 0 \\ k = s - lq}}^{N} (-1)^l \binom{N}{l} \binom{k + N - 1}{N - 1} \\ &= \frac{1}{q^N} \sum_{l=0}^{\lfloor \frac{s}{q} \rfloor} (-1)^l \binom{N}{l} \binom{s - ql + N - 1}{N - 1}. \end{aligned}$$

□

Proposition 6. *Keep the above notation. Let P be the distribution of D_{Kq}. Then, we have*

$$P(D_{Kq} = 0) = \frac{1}{q^N} \sum_{i=0}^{K-1} (-1)^i \binom{N}{i} \binom{(K - i)q + N - 1}{N - 1}.$$

Proof. Since $D_{Kq} = 0$ if and only if $0 \leq S_N \leq Kq - 1$, by Proposition 5 it is enough to compute

$$\sum_{s=0}^{Kq-1} \frac{1}{q^N} \sum_{i=0}^{\lfloor \frac{s}{q} \rfloor} (-1)^i \binom{N}{i} \binom{s - qi + N - 1}{N - 1}. \tag{6}$$

We rewrite the above in terms of i as follows:

$$\binom{N}{i}\sum_{s=iq}^{Kq-1}\binom{s-iq+N-1}{N-1}=\binom{N}{i}\sum_{s=0}^{(K-i)q-1}\binom{s+N-1}{N-1}$$
$$=\binom{N}{i}\binom{(K-i)q+N-1}{N-1}$$

Here, for the last equality, we use

$$\binom{r}{r}+\binom{r+1}{r}+\binom{r+2}{r}+\cdots+\binom{n}{r}=\binom{n+1}{r+1},$$

where $n \geq r \geq 0$. □

D Optimal Hyperparameters Used in the Experiments

Table 7 summarizes the specific hyperparameter pairs (K, r) determined via the preliminary grid search and utilized in our experiments. As observed in the table, there is no obvious regularity or monotonic trend in the optimal values of K and r with respect to the sequence length N or the modulus q. However, as demonstrated by the robustness analysis in Section 5.3, our proposed method achieves consistently high match accuracy across a broad region of the hyperparameter space. Specifically, as long as K is sufficiently large and r is not excessively large, the model maintains stable performance. Therefore, the lack of a strict pattern in these selected optimal values does not pose a critical issue for the general applicability of our method. Nevertheless, further theoretical and empirical analyses are required as future work to systematically derive the optimal K and r for any given problem setting.

Table 7: Optimal hyperparameters (K, r) used for each setting, determined via grid search on a 100K subset. The values are presented separately for **(a)** token embedding and **(b)** angular embedding. Hyphens (-) indicate combinations that were not evaluated.

(a) Token embedding

$N\backslash q$	31	97	257
8	(5, 0.2)	(4, 0.4)	(8, 0.4)
16	(5, 0.4)	(4, 0.4)	(4, 0.1)
32	(5, 0.4)	(8, 0.4)	(9, 0.1)

(b) Angular embedding

$N\backslash q$	97	257	433	3329	42899	974269
16	(5, 0.4)	(6, 0.3)	(6, 0.3)	-	-	-
32	(4, 0.1)	(4, 0.2)	(6, 0.1)	(4, 0.4)	(4, 0.3)	(4, 0.2)
64	(4, 0.3)	(4, 0.3)	(4, 0.3)	(4, 0.3)	(4, 0.3)	(4, 0.2)
128	-	(8, 0.4)	-	(6, 0.4)	(8, 0.4)	(8, 0.3)

E Repeated runs (angular embedding, 100K)

This section reports the repeated runs referred to in Section 5.1. Match accuracy (%) over 10 random seeds (42–51) on the angular setting of Section 5.1 with 100K training samples. Our method uses fixed $K = 6$, $r = 0.5$, which is more conservative than the per-setting pairs in Table 7b. Reported $\pm$ values are standard deviations. Across all nine settings, the worst seed of our method exceeds the best seed of the sparse baseline. For example, at $N = 64$, $q = 433$, the minimum of our method is 12.58% against a sparse maximum of 0.26%.

F Staged training

This section reports the staged-training comparison referred to in Section 5.1. Angular embedding, matched total budget 100K. Staged runs pretrain on the sparse distribution for 75K samples and then continue for 25K samples with either Uniform or Auxiliary targets (weights only; optimizer state is not carried over; learning rate 3×10^{-5}, warmup ratio 0.05). Sparse-only and Auxiliary-only

Table 8: Full 10-seed results (match accuracy %) for angular embedding at 100K samples.

N	q	Ours mean±std	Ours min–max	Sparse mean±std	Sparse min–max
16	97	98.23 ± 0.37	97.78–98.91	30.89 ± 1.99	27.23–34.06
16	257	85.79 ± 1.14	84.37–87.88	11.81 ± 0.98	10.00–13.25
16	433	69.18 ± 3.24	62.61–73.41	7.14 ± 0.72	6.04–8.43
32	97	91.71 ± 1.92	86.37–92.81	3.31 ± 1.42	1.49–5.72
32	257	66.09 ± 3.80	59.93–73.98	1.18 ± 0.55	0.55–1.98
32	433	45.00 ± 3.87	36.76–50.22	0.72 ± 0.43	0.26–1.76
64	97	51.50 ± 4.87	43.35–59.12	0.98 ± 0.08	0.83–1.07
64	257	24.61 ± 4.41	18.71–30.98	0.36 ± 0.05	0.32–0.46
64	433	15.09 ± 1.84	12.58–17.74	0.22 ± 0.03	0.18–0.26

match Table 1b. The (K, r) of Sparse→Auxiliary is that of Auxiliary-only for the same (N, q). The $75\text{K}/25\text{K}$ split is the only schedule reported here.

Sparse pretraining does not surpass Auxiliary-only training. Sparse→Auxiliary improves over Sparse-only on several easier settings (e.g., 63.8% versus 32.2% at $N = 16$, $q = 97$) but remains well below Auxiliary-only, and the gap widens as N grows.

Table 9: Staged-training match accuracy (%) under a matched 100K budget (angular embedding).

N	q	Sparse only	Aux. only (Ours)	Sparse→Unif.	Sparse→Aux.
16	97	32.2	**98.9**	32.6	63.8
16	257	11.8	**93.0**	13.1	38.8
16	433	6.8	**80.7**	8.1	24.1
32	97	4.2	**95.6**	1.1	2.5
32	257	0.6	**73.5**	4.0	21.6
32	433	0.9	**57.5**	0.2	3.8
64	97	0.9	**83.2**	1.0	0.8
64	257	0.3	**43.2**	0.4	0.4
64	433	0.2	**29.0**	0.3	0.3

G Analysis of the τ-accuracy and the α-worst-case τ-bound

This section reports τ-accuracy referred to in Section 5.2. Additional τ evaluations for the large-scale setting of Section 5.2. Unless noted, training uses 1M samples, angular embedding, and fixed $K = 5$, $r = 0.4$ (more conservative than the per-setting best pair in Table 2). Chance τ-accuracy under circular distance is theoretically $\approx 2\tau$ (exact match chance is $1/q$).

Figure 4 plots τ-accuracy against τ for $N = 128$, $q = 974269$, and Table 10 lists the plotted values together with exact match. At $\tau = 0.001$ (allowed circular distance about 974), our method reaches 25.83%, against 2.07% for the sparse baseline and a chance rate of 0.20%. The same separation is already visible at $\tau = 0.0001$ (2.84% versus a chance rate of 0.02%). Exact match is essentially zero at this scale (0.01% for our method). Here τ-accuracy is used for comparison; Section 5.1 remains the exact-match evaluation.

Table 10: τ-accuracy (%) versus τ for $N = 128$, $q = 974269$, 1M samples.

τ	Ours	Sparse	Chance ($\approx 2\tau$)
0 (exact)	**0.01**	0.00	≈ 0.0001
0.0001	**2.84**	0.21	0.02
0.0005	**13.52**	1.00	0.10
0.001	**25.83**	2.07	0.20
0.005	**85.81**	9.52	1.00
0.01	**98.14**	18.59	2.00
0.05	**99.92**	60.99	10.00
0.10	**99.93**	74.11	20.00

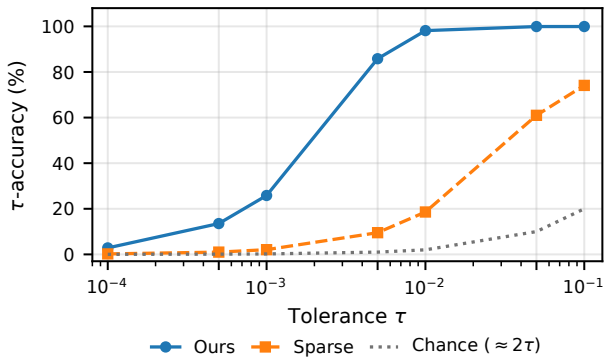


Figure 4: τ-accuracy versus tolerance τ for $N = 128$, $q = 974269$, 1M training samples, and angular embedding ($K = 5$, $r = 0.4$), referred to from Section 5.2. Chance rate under circular distance is $\approx 2\tau$. Exact match ($\tau = 0$) is 0.01% (Ours) and 0.00% (Sparse).

Table 11 inverts the same curve. It reports the smallest τ, on a 66-point grid, at which τ-accuracy reaches a target level, for the $N = 128$, $q = 974269$, 1M run above. Our method attains 90% and 95% at $\tau \leq 0.00607$ and $\tau \leq 0.00786$, about 58 times smaller than the sparse method ($\tau \leq 0.35$ and $\tau \leq 0.45$). It attains 99% by $\tau \leq 0.01364$, while the sparse method and the chance baseline remain at $\tau \leq 0.5$ through that level. Attaining 100% still requires $\tau \leq 0.5$ for our method, so the bound describes how concentrated the prediction error is.

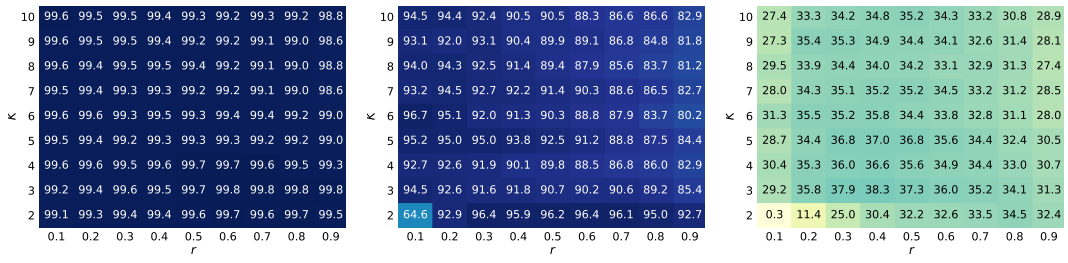
Table 11: α-worst-case τ-bound: smallest τ on a 66-point grid at which τ-accuracy reaches $\alpha\%$ ($N = 128$, $q = 974269$, 1M).

Method	90%	95%	99%	100%
Ours	≤ 0.00607	≤ 0.00786	≤ 0.01364	≤ 0.5
Sparse	≤ 0.35	≤ 0.45	≤ 0.5	≤ 0.5
Chance ($\approx 2\tau$)	0.45	0.475	0.495	0.5

H Extended Heatmaps for Hyperparameter Sensitivity

In Section 5.3 of the main text, we demonstrated the robustness of our proposed method by showing its high average match accuracy across the practical grid search space ($K \in \{4, \ldots, 9\}, r \in \{0.1, \ldots, 0.4\}$). To provide a more comprehensive view of the hyperparameter landscape, this section presents the complete heatmaps expanded over a broader search space: $K \in \{2, \ldots, 10\}$ and $r \in \{0.1, \ldots, 0.9\}$.

Figures 5 and 6 display the full set of grid search results for token embedding and angular embedding, respectively. As discussed in the main text, the proposed method consistently achieves high accuracy across a wide region. A significant decrease in match accuracy is primarily observed only under extreme conditions, such as when K is too small (e.g., $K = 2$, where modulus reduction is insufficient) or when r is excessively large (where the auxiliary task dominates the primary task).

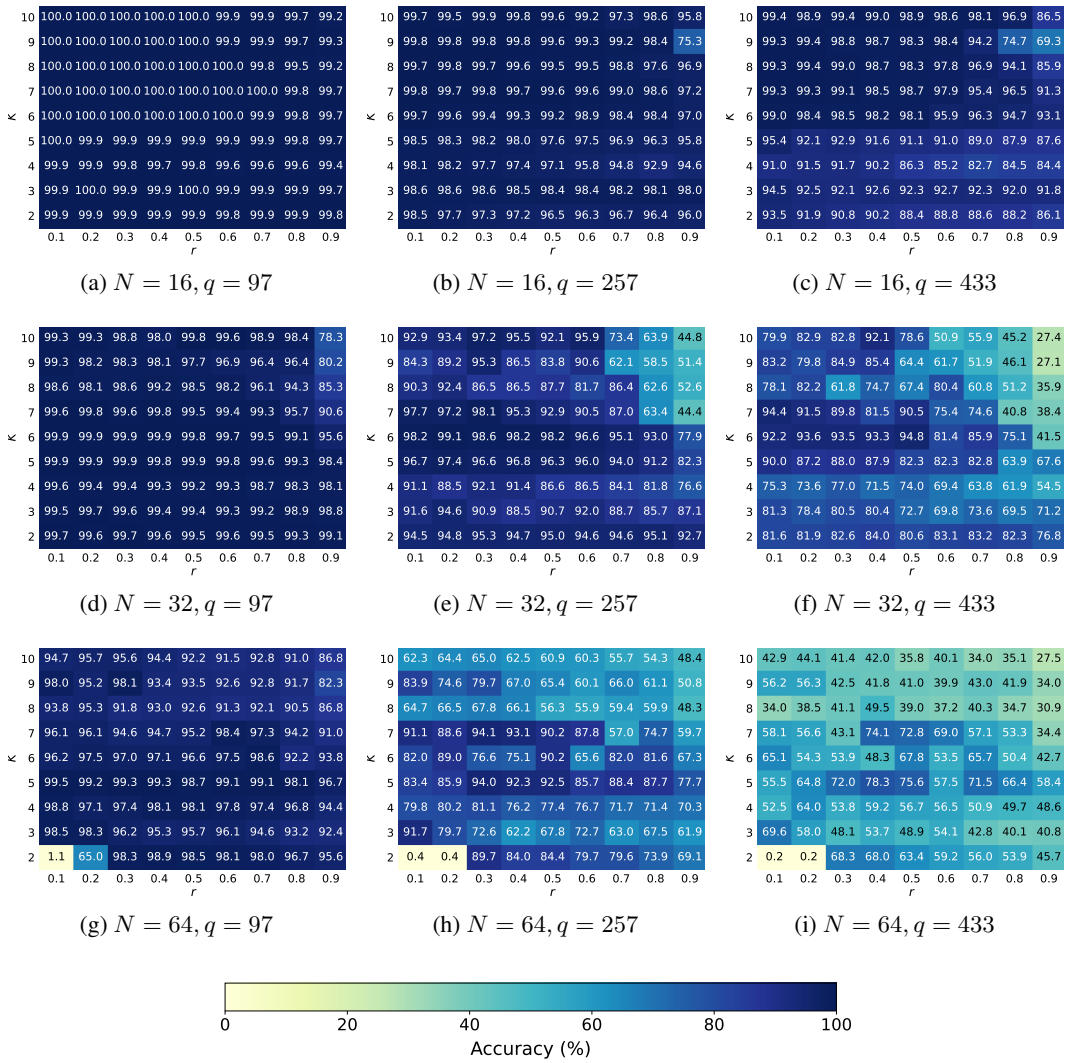


(a) $N = 8, q = 31$ (b) $N = 8, q = 97$ (c) $N = 8, q = 257$

(d) $N = 16, q = 31$ (e) $N = 16, q = 97$ (f) $N = 16, q = 257$

(g) $N = 32, q = 31$ (h) $N = 32, q = 97$ (i) $N = 32, q = 257$

Figure 5: Heatmaps of match accuracy across various combinations of K and r for **token embedding**. The color scale represents the match accuracy (%), which is shared across all nine configurations.



(a) $N = 16, q = 97$ (b) $N = 16, q = 257$ (c) $N = 16, q = 433$

(d) $N = 32, q = 97$ (e) $N = 32, q = 257$ (f) $N = 32, q = 433$

(g) $N = 64, q = 97$ (h) $N = 64, q = 257$ (i) $N = 64, q = 433$

Figure 6: Heatmaps of match accuracy across various combinations of K and r for **angular embedding**. The color scale represents the match accuracy (%), which is shared across all nine configurations.

I Preliminary modular multiplication

Table 12: Multiplication-oriented auxiliary $\bmod q^K$, Sparse, and Uniform at $N = 8$, 1M samples ($K = 2$, $r = 0.5$), referred to from Section 6.

N	q	$\bmod q^K$	Sparse	Uniform
8	31	98.1	99.9	3.19
8	97	1.11	99.9	1.05
8	257	0.38	0.40	0.54

This section reports the preliminary multiplication experiment referred to in Section 6. The auxiliary-task idea is not specific to modular addition and is in principle applicable to multiplication. We therefore replace the auxiliary modulus Kq by a multiplication-oriented target and run a limited check. The runs use a single seed and a small grid, so they indicate applicability rather than a large-scale result. The main contribution of the paper remains modular addition.

The task is $f_q(\mathbf{x}) = (\prod_{i=1}^{N} x_i) \bmod q$ with $x_i \in \{1, \ldots, q-1\}$, evaluated by match accuracy under token embedding. Multiplication depends on the product, whereas the difficulty of addition is governed by the number of wraps $\lfloor (\sum_i x_i)/q \rfloor$. Enlarging the modulus from q to Kq reduces that wrap count for addition, but it does little to simplify a product. A modulus q^K follows the scale of the product, so we keep the input distribution fixed and, with probability r, replace the training target by $(\prod_i x_i) \bmod q^K$.

Table 12 compares $\bmod q^K$, the sparse baseline, and Uniform (uniform inputs, modulus q only) at $N = 8$, 1M samples, $K = 2$, and $r = 0.5$. At $q = 31$, $\bmod q^K$ reaches 98.1%, comparable to the sparse baseline (99.9%) and far above Uniform (3.19%). At $q = 97$ the sparse baseline still reaches 99.9%, while $\bmod q^K$ stays near chance (1.11%). At $q = 257$ every method fails. The positive signal is therefore confined to the easiest setting.

We use token embedding because angular embedding, $x \mapsto (\cos(2\pi x/q), \sin(2\pi x/q))$, is an inductive bias for modular addition, and Saxena et al. [24] likewise use token embedding for multiplication. Under token embedding the classification head for $\bmod q^K$ grows with the vocabulary size q^K, which is impractical for larger q or K. Scaling to larger N and q, and designing a representation suited to multiplication, remain open.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims—specifically that introducing an auxiliary modulus task prevents input distribution shifts and significantly improves both sample efficiency and scalability —are clearly stated in the Abstract and Section 1. Furthermore, these claims are directly supported by the theoretical analysis provided in Section 4 and the experimental results detailed in Section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We explicitly discuss the limitations of our work in Section 6. Specifically, we acknowledge that the proposed method is currently restricted to modular addition tasks. Additionally, we point out that the current reliance on grid search for determining the hyperparameters K and r is a limitation, noting that developing a more efficient optimization method is an important challenge for future work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We provide the full set of assumptions and complete proofs for all theoretical results presented in the paper. Specifically, a proof sketch for Proposition 1 is included in Section 3, while its formal statement (Proposition 3) and full proof are provided in Section A. Furthermore, for the theoretical results introduced in Section 4 (including Proposition 2 and Equation (3)), the complete proof of Proposition 2, the detailed explanation of Equation (3), and the accompanying proofs for Lemma 1, Propositions 5 and 6, and **??** are all provided in Section C. All propositions are properly numbered and cross-referenced.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide all necessary information for independent replication of our experiments. The modular addition dataset requires no external data and can be generated using the exact rules described in Section 5. The core algorithmic modifications—such as the auxiliary loss function and the dual angular embedding—are fully formulated mathematically in Section 4. Training, data generation, and evaluation code is available at the URL in the footnote of the abstract.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Training, data generation, and evaluation code is publicly available at the URL in the footnote of the abstract. Dataset generation and hyperparameters are described in Section 5 and Section D.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We explicitly detail the experimental conditions required to interpret our findings in Section 5. This includes specific training details such as the optimizer (AdamW), learning rate schedules, weight decay, and the sizes of the training and test splits. Importantly, we specify how the hyperparameters K and r were chosen (via grid search on a 100K subset), and we exhaustively document the final chosen values for every experimental configuration in Section D.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: For the angular-embedding match-accuracy comparison with 100K training samples, Section E and Table 8 report the mean, standard deviation, and min–max over 10 random seeds (42–51). The $\pm$ values are standard deviations across seeds, and in all nine settings the worst seed of our method exceeds the best seed of the sparse baseline. The primary tables Tables 1 and 2 remain single-run results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide information regarding the compute resources in Section 5. Specifically, we specify the type of hardware and the amount of memory used for the experiments, and report the estimated execution time per model based on the experimental parameters.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We confirm that our research conforms to the NeurIPS Code of Ethics in every respect. This work is a fundamental study on machine learning algorithms using synthetic datasets and does not involve human subjects, personal data, or any experiments that raise ethical concerns.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: We answer [No] because the paper does not contain a dedicated section discussing broader societal impacts. Our work is foundational research focusing on improving the learning efficiency and stability of deep learning models on modular addition tasks. Although Section 1 briefly mentions the relevance of modular addition to ML-based cryptanalysis for post-quantum cryptography (LWE), our proposed method does not provide a direct, practical tool for attacking deployed cryptographic systems. Therefore, an extended discussion on societal impacts is beyond the scope of this paper.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: We answer [N/A] because our research does not involve the release of data or models that have a high risk for misuse.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [N/A]

Justification: We answer [N/A] because our research does not utilize existing external assets that require license disclosure, such as pre-trained models, specific codebases, or proprietary datasets.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: We answer [N/A] because our paper introduces a new algorithmic methodology rather than releasing distinct new assets such as novel datasets, benchmarks, or pre-trained models.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: We answer [N/A] because our research does not involve crowdsourcing or any research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: We answer [N/A] because our research does not involve human subjects or crowdsourcing.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We answer [N/A] because the core method development in our research does not involve Large Language Models (LLMs) as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.