Learning Large-Scale Modular Addition with an Auxiliary Modulus

Hanato Kikuchi [1] **Ryosuke Masuya** [1] **Kazuhiko Kawamoto** [1] **Hiroshi Kera** [1 2]

Abstract

Learning parity functions, more general 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 $N = 64$ and $q = 974269$, our method trained on 100K samples achieves 97.0% τ-accuracy at $\tau = 0.05$, while the sparse method achieves only 9.5% with the same data size and 93.9% even when extended to 1M samples.[†].

1. Introduction

Despite recent success in learning complex symbolic computational tasks, such as symbolic integration (Charton et al., 2021) and Gröbner basis computation (Kera et al., 2024), and so on (Kera et al., 2026; Alfarano et al., 2024), 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. (2017) and Hahn & Rofin (2024) has proved this for the parity problem (i.e., modular addition with $q = 2$). Larger N, q increase the difficulty further as it incurs more wraps around the modulus. The number of required training samples grows with N and q (Mohamadi et al., 2024); besides, large N necessitates an increased network width (Doshi et al., 2024).

A recent study by Saxena et al. (2025) identified learning modular addition as a key component of attacking the Learning With Errors (LWE) problem, a basis for post-quantum cryptography (US Department of Commerce, 2024), and proposed a learning method that substantially scaled modular addition learning 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 (Shalev-Shwartz et al., 2017). Thus, the key is to limit wrap-around frequency at training time and ease the problem difficulty to make training flow. Saxena et al. (2025) 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 train-

[1]Chiba University, Chiba, Japan [2]National Institute of Informatics, Tokyo, Japan. Correspondence to: Hiroshi Kera <kawa@chiba-u.jp>.

The 3rd *AI for Math Workshop* at the 43rd International Conference on Machine Learning (ICML), Seoul, South Korea, 2026.
Copyright 2026 by the author(s).

[†]https://github.com/hanato-411/auxiliary_modulus.

ing 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. (2025) 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. (2025) 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 of 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 $N = 64$ and $q = 974269$, training on only 100K samples yields 97.0% τ-accuracy (with relative error tolerance $\tau = 0.05$), whereas the sparse method attains 9.5% at the same scale and only 93.9% even with 1M 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 (Nanda et al., 2023), the "Pizza" algorithm (Zhong et al., 2023), and the approximate Chinese Remainder Theorem (McCracken et al., 2026). Additionally, the weights and embeddings of models trained on modular addition also exhibit periodic structures (Power et al., 2022; Liu et al., 2022), and periodic analytical solutions for these weights have been discovered (Gromov, 2023). Furthermore, these periodic structures are observed in other modular arithmetic tasks, such as modular multiplication (Doshi et al., 2024; Furuta et al., 2024). Therefore, helping models learn periodic structures is important. Previous methods, such as angle embeddings (Stevens et al., 2025), 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 (Wei et al., 2022) to decompose problems into steps, treating them as simpler sub-problems. This enables models to solve parity problems that are otherwise difficult to learn (Wies et al., 2023; Kim & Suzuki, 2025). Another effective approach is curriculum learning (Bengio et al., 2009), which gradually increases task difficulty. For parity problems, sparse parity problems (Daniely & Malach, 2020) are utilized as simpler tasks, and applying curriculum learning with them improves both learning accuracy and efficiency on the target problem (Abbe et al., 2023). 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 with modulus of $p = 974269$) than conventional methods (Saxena et al., 2025). However, the method in Saxena et al. (2025) 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 (Caruana, 1997; Ruder, 2017). In mathematical tasks, learning different operations simultaneously improves accuracy. For example, joint training on multiple arithmetic operations enhances performance (Lee et al., 2024). Furthermore, combining simple and difficult tasks reduces model size if they share core building blocks (Both et al., 2026). While Both et al. (2026) 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 $\{0, \ldots, q-1\}^N$ (Saxena et al., 2025). 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 (Saxena et al., 2025). 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 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 (Saxena et al., 2025). 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. (2025) 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:

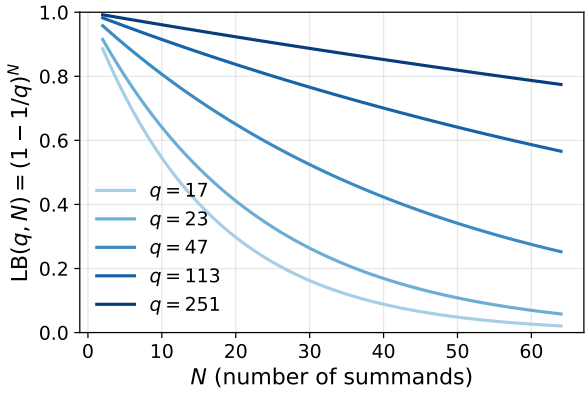
$$y = \begin{cases} f_q(\mathbf{x}) & \text{with probability } 1 - r \\ f_{Kq}(\mathbf{x}) & \text{with probability } r. \end{cases} \tag{2}$$

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.

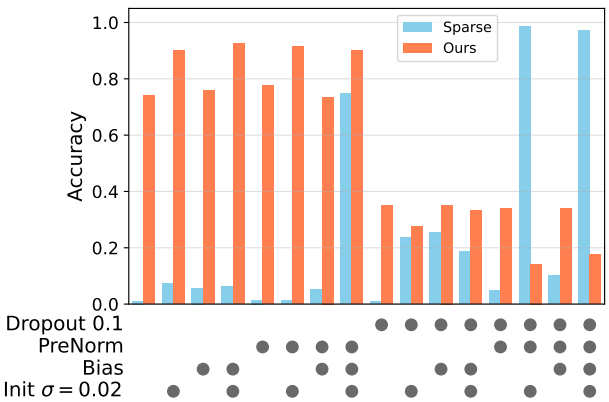
Importantly, the input remains restricted to $\{0, \ldots, q-1\}$, i.e., $\mathbf{x} \in \{0, \ldots, q-1\}^N$, for the auxiliary modulus Kq. The number $D_{Kq}(\mathbf{x})$ of wraps around the (extended) modulus for $\mathbf{x}$ is consequently reduced to

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

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



(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).

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. For simplicity of analysis, we ignore the floor function.

We keep the notation as in Section 4. The (continuous) number of wraps around the modulus is the expectations of the following random variables:

$$X_0 = \frac{\sum_{i=1}^{N} x_i}{q},$$
$$X_1 = (1-r)\frac{\sum_{i=1}^{N} x_i}{q} + r\frac{\sum_{i=1}^{N} x_i}{Kq},$$
$$X_2 = \frac{\sum_{i=1}^{z} x_i}{q},$$

where z is a random variable that represents the number of non-zero entries of an input data, and has a distribution $f_{\mathrm{inv_sqrt}}(z) \propto 1/\sqrt{N - z + 1}$ introduced in Saxena et al. (2025), 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 represents the number of wraps around the modulus, corresponding to the naive method without any training difficulty reduction. The random variables X_1 and X_2 correspond to the proposed method and the sparse method, respectively.

Then, we have the following proposition:

Proposition 2. *Keep the above notation. For X_0 and X_1, we have*

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

For X_2, when N is large enough, we have

$$\mathbb{E}[X_2] \approx \frac{2}{3}\frac{N(q-1)}{2q} = \frac{2}{3}\mathbb{E}[X_0].$$

Proposition 2 shows that all expectations have the same factor $N(q-1)/2q$ ($= \mathbb{E}[X_0]$), 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, without introducing a covariate shift. 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. While this section focuses on the random variable $\sum_{i=1}^{N} x_i/(Kq)$ without the floor function, Section C also examines properties of the original discrete random variable D_{Kq}.

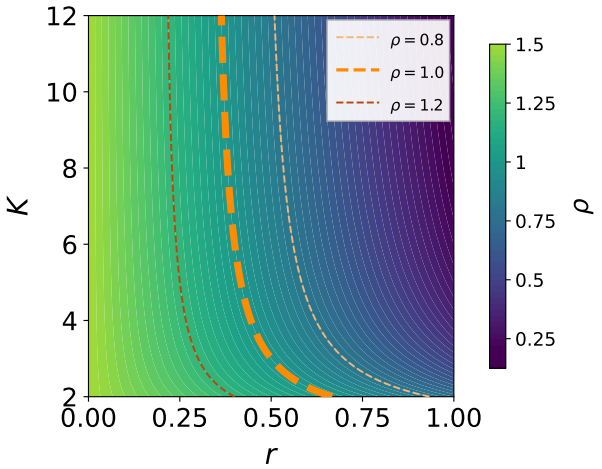


Figure 2. Heatmap of $\rho = \big((1-r)+r/K\big)/(2/3)$ in Equation (4). The orange contour is defined by $\rho = 1$. The range of ρ is from 0.0 to 1.5.

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 scholar 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 (Stevens et al., 2025) to handle the auxiliary modulus Kq.

Approach 2: dual angular embedding. Angular embedding (Stevens et al., 2025) 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 (Saxena et al., 2025) 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 (Saxena et al., 2025) 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 (Saxena et al., 2025). 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. (2025) suggested. The dropout rate was set to 0.0, and positional embeddings were not used. For optimization, the AdamW optimizer (Loshchilov & Hutter, 2019) 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.

Table 1. Comparison of match accuracy (%) between our method and the baseline (Saxena et al., 2025) using 100K and 1M training samples for **(a)** token embedding and **(b)** angular embedding. Note that the baseline is adapted depending on the embedding type, as detailed in the main text. Our method achieves higher match accuracy across various numbers of terms N and moduli q.

(a) Token embedding

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

(b) Angular embedding

		100K		1M	
N	q	Ours	Baseline	Ours	Baseline
16	97	**98.9**	32.2	**99.9**	92.6
	257	**93.0**	11.8	**99.4**	63.8
	433	**80.7**	6.8	**98.5**	45.7
32	97	**95.6**	4.2	**99.6**	81.1
	257	**73.5**	0.6	**88.5**	44.9
	433	**57.5**	0.9	**92.2**	27.7
64	97	**83.2**	0.9	**97.4**	58.7
	257	**43.2**	0.3	**81.1**	23.4
	433	**29.0**	0.2	**53.8**	11.4

Hyperparameter search. The optimal hyperparameter pair ($K \in \{4, \ldots, 9\}$, $r \in \{0.1, \ldots, 0.4\}$) was determined via a grid search on a small dataset ($|\mathcal{D}_{\text{train}}| = 100\text{K}$) and applied across all sample sizes. 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. Section 5.3 provides an ablation study demonstrating strong hyperparameter robustness.

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 $\hat{s} = \frac{\hat{\phi}q}{2\pi} \in [0, q)$, 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., $N = 64, q = 97$).

5.2. Larger-Scale Modular Addition

We next address larger-scale modular addition tasks: $N \in \{16, 32, 64\}$ 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 $\tau \in \{0.01, 0.05\}$) 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 a continuous value converted from the model's output for the input $\mathbf{x}$, and y is the target label of $\mathbf{x}$. Here, $\|\hat{y}-y\|$ represents the distance calculated as $\min(|\hat{y}-y|, q-|\hat{y}-y|)$ to account for the periodic nature of modulo q (i.e., 0 and $q-1$ are near). $\mathbb{1}_S$ is an indicator function of a set S satisfying $\mathbb{1}_S(v) = 1$ if $v \in S$ and 0 otherwise. This metric relaxes the strict exact-match condition by measuring the proportion of predictions that fall within a tolerance margin τq from the true target value.

Results. The experimental results show that the proposed method maintains high sample efficiency and excellent scalability even in large-scale modular addition. As shown in Table 2, even in these larger-scale settings, our method consistently achieves higher τ-accuracy than the baseline method given the same sample size. Specifically, under the condition of 1M samples and $N = 128$, the baseline method shows almost no improvement in τ-accuracy, whereas our method records high τ-accuracy. Furthermore, comparing the results of our method trained on 1M samples with those of the baseline method trained on 10M samples reveals comparable high performance. Notably, in the highly complex setting where $N = 128$, our method using 1M samples outperforms the baseline method using 10M samples.

Table 2. Comparison of τ-accuracy (%) ($\tau \in \{0.05, 0.1\}$) between our method and the baseline (Saxena et al., 2025) using 100K and 1M 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.

		$\tau = 0.1$ Accuracy (%)				$\tau = 0.05$ Accuracy (%)			
		1M		10M		1M		10M	
N	q	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline
16	257	**100**	98.4	**100**	100	**100**	92.1	**100**	99.8
	3329	**100**	98.3	**100**	100	**100**	92.5	**100**	99.9
	42899	**100**	98.6	**100**	100	**99.9**	93.3	**100**	99.9
	974269	**100**	98.4	**100**	100	**100**	93.2	**100**	99.9
32	257	**100**	95.9	**100**	99.9	**99.1**	82.4	**100**	99.6
	3329	**99.9**	95.9	**100**	99.9	**99.2**	82.3	**100**	99.5
	42899	**100**	95.6	**100**	99.9	**99.4**	81.5	**100**	99.5
	974269	**99.7**	95.2	**100**	100	**97.8**	79.9	**100**	99.5
64	257	**99.9**	80.2	**100**	99.8	**98.1**	53.3	**100**	98.8
	3329	**99.8**	70.5	**100**	99.8	**98.0**	43.4	**100**	98.7
	42899	**99.7**	62.1	**100**	99.7	**97.6**	36.5	**100**	98.2
	974269	**99.7**	82.4	**100**	99.7	**97.8**	55.8	**100**	98.5
128	257	**99.1**	10.3	**100**	99.1	**96.4**	5.0	**100**	93.9
	3329	**99.6**	7.6	**100**	99.0	**97.2**	3.9	**100**	93.1
	42899	**98.4**	23.4	**100**	99.1	**94.4**	11.9	**99.6**	93.8
	974269	**99.4**	18.6	**100**	99.2	**97.0**	9.5	**100**	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 E.

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 in Equation (4)—defined as a ratio of expected numbers of wraps around the modulus by the proposed and sparse methods—lies between 0.8 and 1.2. As shown in Section 4.2, in the case when the ratio equals 1.0, the difficulty of a problem under the proposed method theoretically coincides with that under the sparse method. Although high accuracy is not exclusively confined to this box, and the theoretical range does not guarantee uniformly high performance, the region where the ratio approaches 1.0 encompasses the optimal combinations of K and r. Indeed, relatively high-accuracy regions, such as $K \in \{5, \ldots, 7\}$ and $r \in \{0.4, 0.5\}$, largely overlap with this theoretical range.

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 (Saxena et al., 2025), 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

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

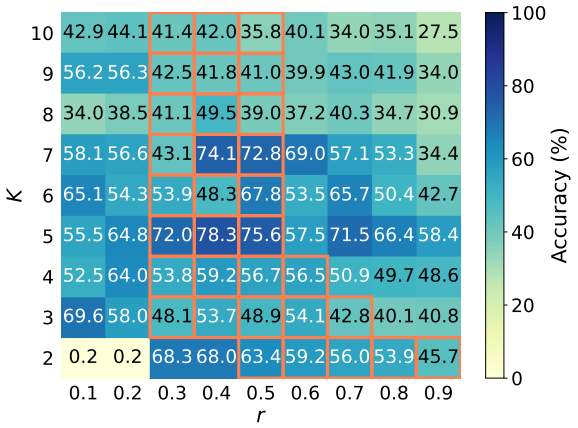


Figure 3. Heatmap of match accuracy for modular addition with angular embeddings ($N = 64, q = 433$). The orange box highlights the region where the ratio ρ in Equation (4) is between 0.8 and 1.2, corresponding to the theoretical difficulty of the prior work shown in Figure 2.

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. Our method is limited to modular addition tasks. In future work, we aim to extend this method to other operations, such as modular multiplication. Additionally, 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 JPMJPR24K, 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

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

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

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 41–48, 2009. URL https://doi.org/10.1145/1553374.1553380.

Both, C., Hoover, B., Strobelt, H., Krotov, D., Weidele, D. K. I., Martino, M., and Dehmamy, N. Small models, smarter learning: The power of joint task training, 2026. URL https://arxiv.org/abs/2505.18369.

Caruana, R. Multitask learning. *Machine learning*, 28(1): 41–75, 1997.

Charton, F., Hayat, A., and Lample, G. Learning advanced mathematical computations from examples. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=-gfhS00XfKj.

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

Doshi, D., He, T., Das, A., and Gromov, A. Grokking modular polynomials, 2024. URL https://arxiv.org/abs/2406.03495.

Furuta, H., Minegishi, G., Iwasawa, Y., and Matsuo, Y. 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.

Gromov, A. Grokking modular arithmetic, 2023. URL https://arxiv.org/abs/2301.02679.

Hahn, M. and Rofin, M. Why are sensitive functions hard for transformers? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 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/.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015. URL https://doi.org/10.1109/ICCV.2015.123.

Kera, H., Ishihara, Y., Kambe, Y., Vaccon, T., and Yokoyama, K. Learning to compute gröbner bases. In *Advances in Neural Information Processing Systems*, volume 37, pp. 33141–33187, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/3a1de90699eec7d7f42c91d81f94af16-Paper-Conference.pdf.

Kera, H., Pelleriti, N., Ishihara, Y., Zimmer, M., and Pokutta, S. 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=bA9rhgWOHk.

Kim, J. and Suzuki, T. 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.

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

Liu, Z., Kitouni, O., Nolte, N. S., Michaud, E., Tegmark, M., and Williams, M. Towards understanding grokking: An effective theory of representation learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 34651–34663, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/dfc310e81992d2e4cedc09ac47eff13e-Paper-Conference.pdf.

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

McCracken, G., Moisescu-Pareja, G., Létourneau, V., Precup, D., and Love, J. 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.

Mohamadi, M. A., Li, Z., Wu, L., and Sutherland, D. J. 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.

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

Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. Grokking: Generalization beyond overfitting on small algorithmic datasets, 2022. URL https://arxiv.org/abs/2201.02177.

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

Saxena, E., Alfarano, A., Wenger, E., and Lauter, K. E. 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.

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

Stevens, S., Wenger, E., Li, C. Y., Nolte, N., Saxena, E., Charton, F., and Lauter, K. E. 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.

US Department of Commerce, N. 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.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., ichter, b., Xia, F., Chi, E., Le, Q. V., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

Wies, N., Levine, Y., and Shashua, A. 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.

Zhong, Z., Liu, Z., Tegmark, M., and Andreas, J. The clock and the pizza: Two stories in mechanistic explanation of neural networks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 27223–27250, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/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. (2025): 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. (2025) 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. (2025) 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 (He et al., 2015).
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. (2025) 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 (Saxena et al., 2025)	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. (2025); 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 X_0, X_1 and X_2 as follows:

$$X_0 = \frac{S_N}{q}, \qquad X_1 = (1-r)\frac{S_N}{q} + r\frac{S_N}{Kq}, \qquad X_2 = \frac{S_z}{q}.$$

Proposition 4 (Proposition 2)**.** *Keep the above notation. For X_0 and X_1, we have*

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

For X_2, when N is large enough, we have

$$\mathbb{E}[X_2] \approx \frac{2}{3}\frac{N(q-1)}{2q} = \frac{2}{3}\mathbb{E}[X_0].$$

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 X_0, X_1 and X_2 are given by

$$\begin{aligned}
\mathbb{E}[X_0] &= \mathbb{E}\left[\frac{S_N}{q}\right] = \frac{N(q-1)}{2q}, \\
\mathbb{E}[X_1] &= \mathbb{E}\left[(1-r)\frac{S_N}{q} + r\frac{S_N}{Kq}\right] = \left((1-r) + r\frac{1}{K}\right)\frac{N(q-1)}{2q}, \\
\mathbb{E}[X_2] &= \sum_{z=0}^{N} \frac{1}{C_N}\frac{1}{\sqrt{N-z+1}}\mathbb{E}\left[\frac{S_z}{q}\right] = \sum_{z=0}^{N} \frac{1}{C_N}\frac{1}{\sqrt{N-z+1}}\frac{z(q-1)}{2q} \\
&= \frac{(q-1)}{2q}\sum_{z=0}^{N}\frac{z}{C_N\sqrt{N-z+1}} = \frac{(q-1)}{2q}\frac{\sum_{z=0}^{N}\frac{z}{\sqrt{N-z+1}}}{\sum_{z=0}^{N}\frac{1}{\sqrt{N-z+1}}}.
\end{aligned}$$

Hence, the statement holds for X_0 and X_1. We prove the statement for X_2 by showing that $\mathbb{E}[X_2] \approx 2\mathbb{E}[X_0]/3$ when N is large enough.

If N is large enough, then the following hold:

$$\begin{aligned}
\sum_{z=0}^{N} \frac{z}{\sqrt{N-z+1}} &= \sum_{k=1}^{N+1} \frac{N-k+1}{\sqrt{k}} = (N+1)\sum_{k=1}^{N+1} \frac{1}{\sqrt{k}} - \sum_{k=1}^{N+1} \sqrt{k} \\
&= (N+1)\sqrt{N} \sum_{k=1}^{N+1} \frac{1}{\sqrt{k/N}} \frac{1}{N} - N\sqrt{N} \sum_{k=1}^{N+1} \sqrt{k/N} \frac{1}{N} \\
&\approx (N+1)\sqrt{N} \int_0^1 \frac{1}{\sqrt{x}} dx - N\sqrt{N} \int_0^1 \sqrt{x} dx \\
&= 2(N+1)\sqrt{N} - \frac{2}{3} N\sqrt{N} \\
&= \frac{4}{3} N\sqrt{N} + 2\sqrt{N}. \\
\sum_{z=0}^{N} \frac{1}{\sqrt{N-z+1}} &= \sum_{k=1}^{N+1} \frac{1}{\sqrt{k}} \\
&= \sqrt{N} \sum_{k=1}^{N+1} \frac{1}{\sqrt{k/N}} \frac{1}{N} \\
&\approx \sqrt{N} \int_0^1 \frac{1}{\sqrt{x}} dx \\
&= 2\sqrt{N}
\end{aligned}$$

Hence, we have

$$\mathbb{E}[X_2] \approx \frac{q-1}{2q} \frac{\frac{4}{3} N\sqrt{N} + 2\sqrt{N}}{2\sqrt{N}} = \frac{q-1}{2q} \left(\frac{2N}{3} + 1 \right) \approx \frac{N(q-1)}{3q}.$$

□

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

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

□

Define a random variable $D_{Kq}(\mathbf{x})$ as in Equation (3). Then, the below proposition holds:

Proposition 6. *We have* $\max\{0, N(q-1)/(2qK)-1\} \leq \mathbb{E}[D_{Kq}] \leq N(q-1)/(2qK)$. *In particular, for* $r \in [0,1]$,

$$\mathbb{E}[r\cdot D_q + (1-r)\cdot D_{Kq}] \leq \frac{N(q-1)}{2q}\left(r + (1-r)\frac{1}{K}\right).$$

Proof. By $\mathbb{E}[S_N/(qK)] = N(q-1)/(2qK)$ and $\max\{0, S_N/(qK)-1\} \leq D_{Kq} \leq S_N/(qK)$, one can prove the statement. □

Proposition 7. *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:

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

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. 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 4 and 5 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).

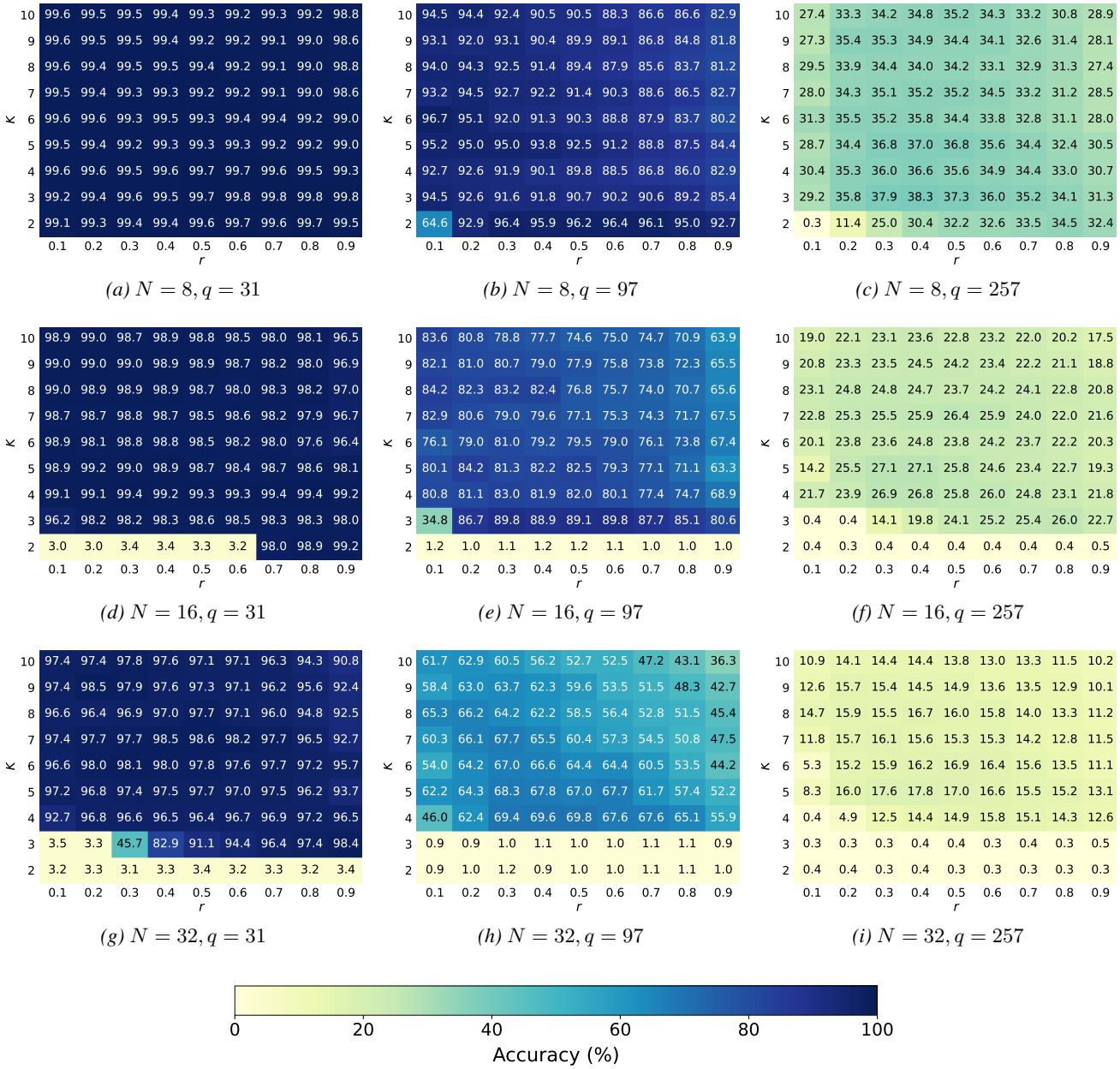
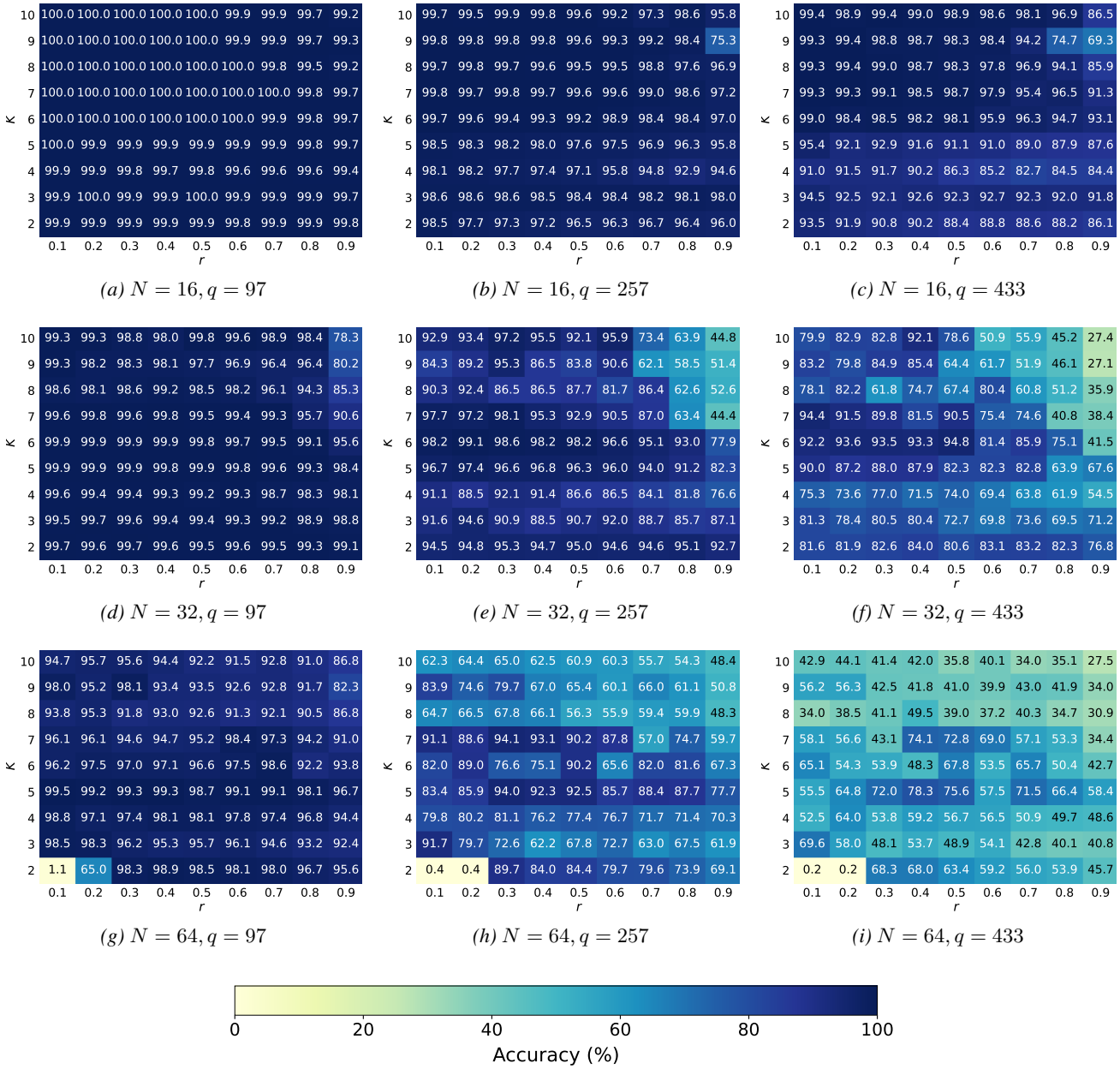


Figure 4. Heatmaps of match accuracy across various combinations of K and r for **token embedding**. The color scale represents the match accuracy (%), which is shared across all nine configurations.



(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 5. 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.