DEL: DIGIT ENTROPY LOSS FOR NUMERICAL LEARNING OF LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Number prediction stands as a fundamental capability of large language models (LLMs) in mathematical problem-solving and code generation. The widely adopted maximum likelihood estimation (MLE) for LLM training is not tailored to number prediction. Recently, penalty-driven approaches, e.g., Number Token Loss and Discretized Distance Loss, introduce an inductive bias of numerical distance but induce over-sharpened and over-flattened digit distributions, respectively. In this paper, we make an in-depth analysis on LLM numerical learning, and show that existing numerical learning methods conceptually follow a criterion-distance formulation, where the criterion term represents optimization pattern and the distance term instills geometric prior. Consequently, we present **D**igit **E**ntropy **L**oss (DEL) for auto-regressive numerical learning, which reformulates the conventional unsupervised entropy optimization in three key designs: leveraging digit conditional probability and binary cross-entropy to guide the entropy optimization into a supervised manner; deprecating the distance term to bypass the issue of numerical distance; and generalizing the integer-based numerical learning to floating-point number optimization, enabling more accurate number prediction. Our DEL formulation can incorporate integers, decimals, and decimal points, expanding the learning objective from a single digit to the floating-point number domain. Experiments conducted on seven mathematical reasoning benchmarks with four representative LLMs, including CodeLlama, Mistral, DeepSeek, and Qwen-2.5, demonstrate that DEL consistently outperforms its counterparts in both overall prediction accuracy and numerical distance. Source codes and trained models will be publicly available.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable natural language processing capabilities, gradually evolving from the handling of simple human-computer dialogues to complex advanced reasoning, such as scientific computation (Wang et al., 2024a; Tong et al., 2024; Tang et al., 2024) and code generation (Roziere et al., 2023; Yue et al., 2024; Toshniwal et al., 2024). As numbers are ubiquitous used in our daily life, the ability to understand, reason with and accurately generate numerical information has emerged as a foundational capability of LLM, serving as the bedrock for quantitative analysis, logical deduction, and the stability of executing algorithms within the aforementioned domains.

Since the pioneering work in (Brown et al., 1992), a common approach to optimize language models is maximum likelihood estimation (MLE), which maximizes the likelihood of conditional probabilities of the generated language sequence. However, it has been found that optimizing language models using MLE is sub-optimal as the zero-avoiding property of MLE forces the model to assign non-trivial probabilities to any observed tokens in the training data. This will lead to issues such as text degeneration (Holtzman et al., 2020), deviation from human-like text (Zhang et al., 2023), and data void overestimation (Ji et al., 2023). To alleviate such problems, later representative methods (Ji et al., 2023; Zhang et al., 2023; Ren et al., 2024) have considered the use of new distance alternatives, which show effectiveness in improving the LLM prediction of general texts. However, their improvements in number prediction remain limited, as these approaches are designed for general textual sequences to enhance semantic coherence and generation diversity and neglect the characteristics of numerical optimization that prioritize precise digit outputs and deterministic numerical representations.

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

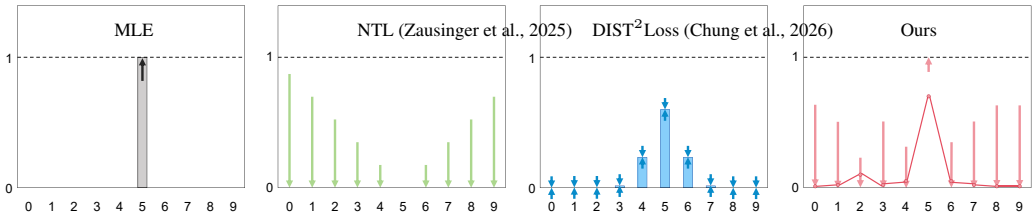


Figure 1: Comparison of the optimization objectives of MLE, NTL (Zausinger et al., 2025), DIST^{2}Loss (Chung et al., 2026), and our proposed DEL. In the illustration, 5 is the target token.

Recent advances in numerical learning (Zausinger et al., 2025; Chung et al., 2026) offer a promising alternative by introducing a penalty-driven approach. By adding a penalty term to the cross entropy loss to measure the proximity between digits, the supervision signals carried by digits can provide positive guidance to LLMs. As shown in Fig. 1, NTL (Zausinger et al., 2025) explicitly suppresses the probability of non-target tokens to nearly 0, while DIST^{2}Loss (Chung et al., 2026) enforces a smooth objective distribution. As we will analyze later, both NTL and DIST^{2}Loss conceptually follow a criterion-distance formulation, where the criterion term provides the optimization pattern for the LLMs to learn and the distance term determines the degree of penalty imposed on each digit probability. However, these two losses have some limitations. *Firstly, they heavily rely on prior knowledge, i.e., the definition of the distance term.* For example, if the target digit is 3, predicting 7 is further away than predicting 4, and therefore a greater penalty will be imposed. A rigorous constraint as in (Zausinger et al., 2025) can easily lead to over-sharpened digit distribution, while a soft constraint as in (Chung et al., 2026) can produce greater uncertainty in generating digits. In addition, the semantic information of numbers is multifaceted and cannot be fully captured by numerical distance alone. *Secondly, they lack holistic number awareness.* Specifically, NTL (Zausinger et al., 2025) operates within a single-digit optimization paradigm, which limits its ability to model the global numerical structure of multi-digit numbers. DIST^{2}Loss (Chung et al., 2026) introduces the digital place weighting for integers, but the discontinuous penalty between the integers and the decimals still persists since the decimal is treated as a separate integer.

In this paper, we propose **D**igit **E**ntropy **L**oss (DEL) for auto-regressive numerical learning. Inspired by the phenomenon revealed in (Wang et al., 2025), high entropy tokens are mostly logic connectors and transitions, while low entropy tokens exhibit determinacy. We argue that highly deterministic generation is fundamentally crucial for numerical tokens in language models. This is because as a discrete and precise symbolic system, the semantics of numbers come from their magnitude, digit sequence, and proportional relationships. Moreover, numerical determinacy serves as an indispensable prerequisite for reliable LLM deployment in downstream tasks. Guided by these insights, we shift our attention towards entropy minimization. However, conventional entropy minimization may struggle in multiple local optima because any one-hot vector is one of the solutions. To address this issue, we reformulate the conventional entropy optimization with the following three key designs. First, we propose to use the combination of digit conditional probability and binary cross-entropy as the new criterion term, where we particularly concentrate on the subspace of digits. Second, we deprecate the distance term because it does not strictly follow the semantic representations of digits learned by LLMs from massive real-world corpora. Finally, we generalize the numerical learning to floating-point number optimization, which recognizes all floating-point numbers in the ground-truth token sequences during training, enabling the language models with holistic number awareness.

To evaluate our proposed method, we apply DEL to several popular LLMs, including CodeLlama, Mistral, DeepSeek-Math, and Qwen-2.5. We demonstrate that DEL is capable of boosting number prediction performance in both Chain-of-Thought (CoT) reasoning and Program-of-Thought (PoT) code generation. The experiments on seven mathematical reasoning benchmarks show that DEL achieves consistent accuracy gains and lower numerical distance over its state-of-the-art counterparts.

2 Related Work

Probabilistic Language Models. Probabilistic language models have long been used in Natural Language Processing (NLP), e.g., neural machine translation (Och, 2003; Sennrich et al., 2016), speech recognition (Graves & Jaitly, 2014; Radford et al., 2023), and optical character recognition

(OCR) (Graves et al., 2006; Li et al., 2023), etc. This language modeling approach can be dated back to pioneer work (Brown et al., 1992) in 1992. By modeling a series of conditional probabilities over the sequences of symbols pertaining to a language, researchers have witnessed extraordinary reasoning capability emerging from the language models after training on large-scale humanity data. Then, the emergence of LSTM (Hochreiter & Schmidhuber, 1997) and transformers (Vaswani et al., 2017) greatly enhanced the ability of language models to model long sequences. Subsequently, word embedding techniques have evolved from learning static representations for words (Bengio et al., 2003; Mikolov et al., 2013) to context-aware, dynamic representations like BERT (Devlin et al., 2019). Another interesting approach is prompting method. With some task-specific exemplars, language models (Brown et al., 2020) can better perform a variety of tasks and even generate the chain-of-thought (CoT) (Wei et al., 2022; Mitra et al., 2024; Yao et al., 2023), which shows better performance in math word problems. Recently, Program-of-Thought (PoT) (Chen et al., 2023) and Program-aided Language Models (PAL) (Gao et al., 2023) have explored code generation to overcome the numerical calculation errors and the inefficiency of multiple iterations.

Probabilistic Sequence Optimization. When generating a sequence, the language model performs next token prediction in an autoregressive manner. After gathering the inference results, the maximum likelihood estimation is used to optimize the sequence probabilities. The idea of improving such sequence probability is not new. For example, MixCE (Zhang et al., 2023) combines the reversed cross-entropy to help the language model to generate more like humans. TaiLr (Ji et al., 2023) utilizes the total variation distance to reduce the penalization intensity of low probability predictions. Another attempt is EMO (Ren et al., 2024), which considers the Earth Mover Distance (EMD) to drive the differentiable upper bound of EMD. Recently, optimization for numerical sequences has received increasing attention, as predicting numerical sequences is the core of many reasoning tasks, e.g., arithmetic (Yuan et al., 2023), mathematical reasoning (Cobbe et al., 2021; Hendrycks et al., 2021b), code generation (Roziere et al., 2023; Yue et al., 2024), etc. It is also a fundamental problem in some vision language models (VLM) (Liu et al., 2023; Chen et al., 2024; Wang et al., 2024b), where the language model begins to take on some object-related visual grounding and clue inferring. To optimize the numerical sequence, a successful approach, namely Number Token Loss (NTL) (Zausinger et al., 2025), has been found useful to mathematical reasoning. NTL incorporates digit distance into the objective function, thereby endowing the language models with the awareness of digit prediction errors. More recently, DIST^{2}Loss (Chung et al., 2026) proposes to mimic a digit-distance-driven soft target, where a KL-divergence is imposed between the predict sequence probabilities and the soft targets. Despite noticeable improvement over the maximum likelihood estimation, they are essentially based on a heuristic definition of numerical distance.

It has been a common sense that standard CE implicitly reduces overall ambiguity via the global softmax normalization, but this implicit effect is inefficient within the digit subspace since digit tokens account for only a tiny fraction of the full vocabulary. Motivated by the aforementioned methods, we study the numerical learning in language models with in-depth analyses on the existing approaches, exploring effective means to adapt the numerical nature.

3 Numerical Learning in Language Models

Background. In reasoning tasks that involve numeric outputs, the language model $\mathbf{M}$ with parameters Θ typically predicts numbers in an auto-regressive, token-by-token manner (Raffel et al., 2020; Radford et al., 2019; Touvron et al., 2023a). Let X be a language sequence of length T. When predicting the t-th token, the language model outputs logits $z_t = \mathbf{M}_\Theta(\cdot|x_{<t})$, and converts it to the conditional probability distribution $\mathbf{p}(\cdot|x_{<t}) = \mathcal{S}(z_t)$ over the vocabulary V, where $\mathcal{S}(\cdot)$ is the Softmax function. We call the probability $\mathbf{p}(\cdot|x_{<t})$ **the full space probability** since it represents the probability of taking x_t as the predicted token over the full vocabulary token space V. During training, all the probabilities are gathered to a large probability sequence $\mathbf{p}$. Here, we only discuss the digit tokens and omit the non-digital tokens for brevity. Then, the standard MLE is used to minimize the distance between the probability sequence $\mathbf{p}$ and the ground-truth digit token sequence $\mathbf{g}$. This solution, also known as the language modeling loss, has been dominantly adopted in various popular language models, including GPT (Radford et al., 2019; Brown et al., 2020; Achiam et al., 2023), Llama (Touvron et al., 2023a;b; Roziere et al., 2023), Qwen (Bai et al., 2023; Team et al., 2024; Qwen et al., 2025), Mistral (Jiang et al., 2023), Vicuna (Zheng et al., 2023), etc.

To facilitate numerical learning, the NTL method (Zausinger et al., 2025) adds a penalty term focusing on the digits during training, which can be written as:

$$\mathcal{L}_{NTL} = \underbrace{\mathcal{H}(\mathbf{p}, \mathbf{g})}_{\text{cross-entropy}} + \alpha \sum_{\mathbf{g}_t \in V_d} \hat{\mathbf{p}}(\cdot|x_{<t}) ||V_d - \mathbf{g}_t||_1, \tag{1}$$

where $V_d = \{0, 1, \cdots, 9\}$ is the digit vocabulary, $\hat{\mathbf{p}}(\cdot|x_{<t}) = \mathcal{S}(\hat{z}_t) = \mathcal{S}(\mathbf{M}_\Theta(\cdot|x_{<t})|_{V_d})$ denotes **the digit conditional probability** over V_d, $\mathbf{g}_t$ denotes the t-th ground-truth token, and α is a hyperparameter balancing the two terms. Due to the introduction of penalty term, the predicted digit probabilities will shift towards the target digit distribution. Note that we use the NTL with the Wasserstein-1 Distance (NTL-WAS) in the above equation since it generally achieves better accuracy as evidenced by (Zausinger et al., 2025) both theoretically and experimentally.

Another recent study DIST^{2}Loss (Chung et al., 2026) suggests distance modeling in loss functions. It first formulates a target distribution $\mathbf{q}_t$:

$$\mathbf{q}_t = \mathcal{S}(-\rho(V_d, \mathbf{g}_t), \tau), \quad \mathbf{g}_t \in V_d, \tag{2}$$

where $\mathcal{S}(\cdot, \tau)$ denotes the generalized softmax function with temperature τ, ρ is a pre-defined distance metric. Then, a KL-divergence loss is imposed between the predicted digit probability $\hat{\mathbf{p}}(\cdot|x_{<t})$ and the $\mathbf{q}_t$ defined above, which can be written as:

$$\mathcal{L}_{dist} = \mathcal{H}(\mathbf{p}, \mathbf{g}) + \alpha \sum_{\mathbf{g}_t \in V_d} \mathcal{L}_{\text{KL}}(\mathbf{q}_t \parallel \hat{\mathbf{p}}(\cdot|x_{<t})). \tag{3}$$

A Closer Look at Numerical Learning. It can be deduced that the above two losses are conceptually equivalent to the following form of loss function:

$$\mathcal{L}_N = \mathcal{H}(\mathbf{p}, \mathbf{g}) + \alpha \sum_{\mathbf{g}_t \in V_d} \underbrace{C(\hat{\mathbf{p}}(\cdot|x_{<t}))}_{\text{criterion term}} \cdot \underbrace{D(V_d, \mathbf{g}_t)}_{\text{distance term}}, \tag{4}$$

where $C(\hat{\mathbf{p}}(\cdot|x_{<t}))$ optimizes the criterion between the predicted digit probability $\hat{\mathbf{p}}(\cdot|x_{<t})$ and the ground-truth digit $\mathbf{g}_t$; $D(V_d, \mathbf{g}_t)$ measures the distance between the digit vocabulary V_d and $\mathbf{g}_t$. Specifically, NTL (Eq. 1) attempts to minimize the probabilities for the non-target tokens by defining $C(\hat{\mathbf{p}}(\cdot|x_{<t})) = \hat{\mathbf{p}}(\cdot|x_{<t})$. Then, the distance term $D(V_d, \mathbf{g}_t)$ guides the model to capture how far each numeric token is from its ground-truth value. In DIST^{2}Loss, expanding the penalty term $\mathcal{L}_{\text{KL}}(\mathbf{q}_t \parallel \hat{\mathbf{p}}(\cdot|x_{<t}))$ and ignoring the zero gradient term, one can get:

$$\mathcal{L}_{\text{KL}}(\mathbf{q}_t \parallel \hat{\mathbf{p}}(\cdot|x_{<t})) = \mathcal{H}(\mathbf{q}_t, \hat{\mathbf{p}}(\cdot|x_{<t})) = -\mathbf{q}_t \cdot \log \hat{\mathbf{p}}(\cdot|x_{<t}), \tag{5}$$

which implies that $C(\hat{\mathbf{p}}(\cdot|x_{<t})) = -\log \hat{\mathbf{p}}(\cdot|x_{<t})$ and $D(V_d, \mathbf{g}_t) = \mathbf{q}_t$.

It can be seen that both losses penalize the prediction errors for the non-target digit tokens. Comparing the two losses, we see that the key difference lies in the distance term. In NTL, the distance term penalizes the criterion term by the l_1 distance between the digit vocabulary and the ground-truth digit, while the distance term in DIST^{2}Loss weights the distance with a soft distance-based distribution. More analyses of numerical learning can be found in **Appendix A**.

Conclusion. In spite of the differences between NTL and DIST^{2}Loss, it can be concluded that they share the same optimization paradigm, which we call *criterion-distance numerical learning*, as laid down in Eq. 4. Both NTL and DIST^{2}Loss combine a token-probability-related criterion term with a manually defined numerical distance term, and inject geometric proximity inductive bias into the model by penalizing non-target digits proportional to their numerical distance from the ground truth. This paradigm also carries notable limitations in practical design: First, the distance term is built upon manually defined geometric prior about numerical proximity. This heuristic geometric constraint may not always fully align with the semantic representations of digits that LLMs learn from large-scale real-world corpora, as verified by our analysis in Sec. 4. A relatively strict distance constraints tend to yield over-sharpened digit distributions, while softer constraints may introduce unnecessary generation uncertainty; both deviate from the ideal deterministic behavior required for high-quality numerical generation. Second, the existing paradigm is primarily framed around single-digit token-level optimization, but lacks of holistic awareness of multi-digit numbers, resulting in discontinuous penalty signals when processing floating-point numbers.

4 DIGIT ENTROPY LOSS

In this section, we introduce **D**igit **E**ntropy **L**oss (DEL) for autoregressive numerical learning. Grounded in the entropy minimization principle for deterministic digit generation, DEL reformulates conventional unsupervised entropy optimization into a supervised paradigm via digit conditional probability and binary cross-entropy, deprecates the explicit distance term to retain model adaptability, and extends the optimization scope to floating-point numbers for holistic numerical awareness.

From Criterion-Distance to Digit Entropy Optimization. Eq. 4 shows that the criterion term is the key to the entire formulation since it provides the gradient flow for the language model to learn. We draw inspiration from (Wang et al., 2025), which reveals that the high entropy tokens acts as pivotal decision points in CoTs, whereas the low entropy tokens indicate determinacy. We argue that highly deterministic number generation is crucial for language models as it is an indispensable prerequisite for reliable LLM deployment in downstream tasks. An intuitive way to minimize the digit entropy is to take the digit conditional entropy as the new criterion term for numerical learning, which can be written as:

$$\mathcal{L}_{DE} = \mathcal{H}(\mathbf{p}, \mathbf{g}) + \alpha \sum_{\mathbf{g}_t \in V_d} \mathcal{H}(\hat{\mathbf{p}}(\cdot|x_{<t})) = \mathcal{H}(\mathbf{p}, \mathbf{g}) - \alpha \sum_{\mathbf{g}_t \in V_d} \hat{\mathbf{p}}(\cdot|x_{<t}) \log \hat{\mathbf{p}}(\cdot|x_{<t}), \tag{6}$$

where $\mathcal{H}(\mathbf{p}, \mathbf{g})$ is the standard cross-entropy loss, and $\mathcal{H}(\hat{\mathbf{p}}(\cdot|x_{<t}))$ is the Shannon entropy (Shannon, 1948) of the conditional probability distribution of digit tokens. However, minimizing the digit conditional entropy is unsupervised, which may yield many trivial solutions. Consequently, this objective encourages the model to produce sharper probability distributions, but it cannot ensure that the probability mass is optimized to the target digits.

Digit Entropy Loss. To avoid the confusion of infinite solutions in digit entropy optimization, we propose to jointly optimize the product of the digit conditional probability and the binary cross-entropy, which is given by:

$$\begin{aligned} \mathcal{L}_{DEL} &= \mathcal{H}(\mathbf{p}, \mathbf{g}) + \alpha \sum_{\mathbf{g}_t \in V_d} \hat{\mathbf{p}}(\cdot|x_{<t}) \mathbf{BCE}(\dot{\mathbf{p}}(\cdot|x_{<t}), y_t) \\ &= \mathcal{H}(\mathbf{p}, \mathbf{g}) - \alpha \sum_{\mathbf{g}_t \in V_d} \hat{\mathbf{p}}(\cdot|x_{<t}) \Big(y_t \log \dot{\mathbf{p}}(\cdot|x_{<t}) + (1 - y_t) \log \big(1 - \dot{\mathbf{p}}(\cdot|x_{<t})\big) \Big), \end{aligned} \tag{7}$$

where $y_t \in \mathbb{R}^{10}$ is the one-hot label vector of $\mathbf{g}_t$, and $\dot{\mathbf{p}}(\cdot|x_{<t}) = \sigma(\mathbf{M}_{\Theta}(\cdot|x_{<t})|_{V_d})$ is the digit conditional probability for binary classification, calculated by the sigmoid function σ. In the formulation, the penalty term of DEL has a global minima at $\hat{\mathbf{p}}(\mathbf{g}_t|x_{<t}) = 1$, and 0 for any $x_t \neq \mathbf{g}_t$. Compared to NTL, DEL sharpens the learned digit distribution in a moderate manner. Specifically, the product $\hat{\mathbf{p}}(\cdot|x_{<t}) \cdot \mathbf{BCE}(\cdot)$ imposes a soft constraint: the BCE term's gradient weakens as non-target probabilities decrease, i.e., $\log(1 - \dot{\mathbf{p}}(v|x_{<t})) \to 0$, while the penalty term of NTL is a constant based on numerical distance, which sustainably pushes the non-target probabilities to 0.

Deprecation of the Distance Term. The distance term in previous works (Zausinger et al., 2025; Chung et al., 2026) acts for two purposes: weighting the gradients and mimicking the proximity between digits. It imposes a geometric prior, which specifies that the language model must conform to numerical distance. However, for many natural language scenarios, digits serves as symbols, e.g., dates, addresses, serial identities, and convey degree, etc. It has been found that digits learned by LLMs exhibit an entangled representation, blended by both string-like and numerical representations (Marjieh et al., 2025). Shao et al. (Shao et al., 2025) observed Benford's Curse in the first place of numbers caused by digit bias in web-collected corpora. To investigate the semantic structure of digits, we analyze the digital embeddings learned by LLMs. For each digit $x_i \in V_d$, we feed it to an LLM and obtain an embedding e_i from the last hidden state of transformer. Fig. 2 **Left** shows the t-SNE visualization (Van der Maaten & Hinton, 2008) of digit token embeddings on 2D projection surface. It can be seen that the digit distribution is scattered with weak numerical order. For instance, the digit 0 is not the closest to 1, but close to 8. Fig. 2 **Middle** and **Right** show the differences between the digit token embedding similarity and the ideal numerical distance (in l_1-norm). It is worth noting that these phenomena are hold for LLMs, though the distribution of digits might differ. This indicates that numerical distance may not be fully aligned with the semantic information learned by LLMs. In other words, for example, if an LLM treats the digits 2 and 3 as strings, the summation of the two digits

Table 1: The optimization objectives of NTL, DIST^{2}Loss and our DEL.

Method	criterion	distance	Optimal solutions
NTL	digital probability $\hat{\mathbf{p}}(\cdot\|x_{<t})$	$\|\|V_d - \mathbf{g}_t\|\|_1$	$\hat{\mathbf{p}}(v\|x_{<t}) \to 0$
DIST^{2}Loss	digital log-probability $\log \hat{\mathbf{p}}(\cdot\|x_{<t})$	soft target $\mathbf{q}_t$	$\hat{\mathbf{p}}(\cdot\|x_{<t})) \to \mathbf{q}_t$
DEL (Ours)	$\hat{\mathbf{p}}(\cdot\|x_{<t})\mathbf{BCE}(\dot{\mathbf{p}}(\cdot\|x_{<t}), y_t)$	-	$\hat{\mathbf{p}}(\cdot\|x_{<t}) \to 0$ or $\dot{\mathbf{p}}(\cdot\|x_{<t}) \to y_t$

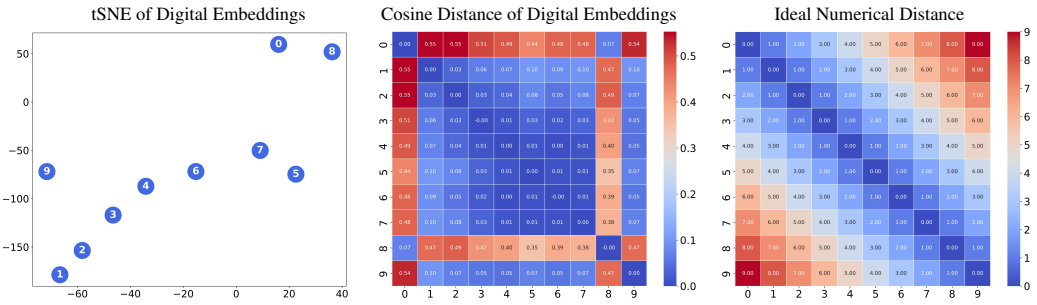


Figure 2: Visualization of digit token embeddings and their similarity structure on Qwen-2.5-7B (Qwen et al., 2025). **Left**: t-SNE 2D projection (Van der Maaten & Hinton, 2008) of digit token embeddings. **Middle**: Cosine distance heatmap of digit embeddings. **Right**: Ideal numerical distance heatmap (Zausinger et al., 2025) for comparison.

may be 23 rather than 5. Although the manually designed numerical distance is effective compared to the vanilla CE, we argue that such explicit prior constraints inevitably introduce over-regularization, limiting the model's flexibility in fitting diverse data distributions. Thus, we propose to deprecate the distance term, endowing LLMs with adaptive capability of learning the right digit probability distribution. Our empirical results later provide the primary support for our design.

Tab. 1 summarizes the optimization objectives of NTL, DIST^{2}Loss and our proposed DEL. We can see that both NTL and DIST^{2}Loss rely on the distance term to shape the digit distribution, while our DEL derives from entropy optimization, which is free from the manually defined distance term. The introduced binary digit probabilities also offer a new optimization path for numerical learning.

Floating-point Number Optimization. In real-world scenarios, numbers are often dominated by multi-digits. The most common examples are multi-digit integers and floating-point numbers. Given a numerical token series:

$$\mathbf{g} = \mathbf{g}_0^L \mathbf{g}_1^{L-1} \cdots \mathbf{g}_L^0 . \mathbf{g}_{L+2}^{-1} \mathbf{g}_{L+3}^{-2} \cdots \mathbf{g}_{L+M+1}^{-M}, \tag{8}$$

which represents a multi-digit floating point number $\mathbf{g}$, with the highest integer place $\mathbf{g}_0^L$, the unit place $\mathbf{g}_L^0$, the tenths place $\mathbf{g}_{L+2}^{-1}$, and the lowest decimal place $\mathbf{g}_{L+M+2}^{-M}$. It has been discussed in (Chung et al., 2026) that extending DIST^{2}Loss to multi-digit integers may face the problem of credit assignment. This is because the global distance metric is the result of the collective action of all tokens and cannot be simply broken down into the independent distance of each token. To alleviate this issue, DIST^{2}Loss integrates the place value weighting to allocate more penalty to higher place digits. However, this strategy fails when dealing with floating-point numbers, because the decimal will be treated as a new integer, resulting in discontinuous penalty. We therefore propose floating-point number optimization, which delivers larger penalty for integers, and a small penalty for decimals. We suggest a linear penalty for integer places and an exponential decay for decimals:

$$u(t) = \begin{cases} ki + 1, & \text{for } \mathbf{g}_t^i,\ i \geq 0, \\ \beta^i, & \text{for } \mathbf{g}_t^i,\ i < 0, \end{cases} \tag{9}$$

where parameters $k, \beta > 0$ control the penalty amplitude for each token. The visual comparison of $u(t)$ in DIST^{2}Loss and DEL can be seen in Fig. 4 of **Appendix B**. Finally, we generalize DEL from single-digit optimization to floating-point number optimization by rewriting Eq. 7 as follows:

$$\mathcal{L}_{DEL} = \mathcal{H}(\mathbf{p}, \mathbf{g}) + \alpha \sum_{\mathbf{g}_t \in V_d} u(t) \hat{\mathbf{p}}(\cdot|x_{<t}) \mathbf{BCE}(\dot{\mathbf{p}}(\cdot|x_{<t}), y_t). \tag{10}$$

5 EXPERIMENTS

In this section, we conduct extensive experiments to demonstrate the performance of the proposed DEL. We apply DEL to several recently popular LLMs, and evaluate it together with state-of-the-art losses on the classic mathematical reasoning and code generation tasks.

5.1 EXPERIMENTAL SETUP

Baseline and Compared Methods. We conduct experiments using four widely used open-source LLMs, including CodeLlama (Roziere et al., 2023), Mistral (Jiang et al., 2023), Qwen2.5 (Qwen et al., 2025), and DeepSeek-Math-instruct (He et al., 2025b). MLE is used as the baseline loss. To make the comparison clean and fair, all experimental settings irrelevant to the loss function remain unchanged for a given LLM. We compare DEL with the following representative methods: MixCE (Zhang et al., 2023) and EMO (Ren et al., 2024), which aim at improving language understanding on general texts, and the recently developed numerical learning methods NTL (Zausinger et al., 2025) and DIST^{2}Loss (Chung et al., 2026). In addition, we also evaluate 12 off-the-shelf open-source LLMs, which have made remarkable progress in mathematical reasoning.

Datasets. We evaluate DEL on seven widely used mathematical reasoning benchmarks: 1) high quality grade school math problems dataset GSM8K (Cobbe et al., 2021); 2) multi-level difficulty math competition dataset MATH (Hendrycks et al., 2021b); 3) arithmetic math word problem dataset SVAMP (Patel et al., 2021); 4) simultaneous equations dataset SimulEq (Koncel-Kedziorski et al., 2016); 5) algebraic word problems with rationales AQuA (Ling et al., 2017); 6) general college entrance exams SAT-Math (Zhong et al., 2024); and 7) college math problems dataset MMLU (Hendrycks et al., 2021a). We report zero-shot in-context-learning task accuracy and the mean accuracy (mACC) over the evaluated benchmarks. To evaluate the applicability, we adopt CoT reasoning for AQuA, SAT-Math, and MMLU, while we use program-of-thought (PoT) for code generation on GSM8K, MATH, SVAMP, and SimulEq. Training is conducted on the large-scale high-quality math instruction tuning dataset MathInstruct (Yue et al., 2024), which consists of 13 math datasets covering a broad variety of math fields and complexity levels.

Implementation Details. We adopt the Huggingface transformer library (Wolf et al., 2020) to conduct the experiments. We train all the models for 2 epochs. The AdamW (Loshchilov & Hutter, 2019) is adopted as the optimizer with bfloat16 precision enabled. We use a learning rate of 2e-5 for CodeLlama, and 5e-6 for Mistral, Qwen2.5, and DeepSeek-Math-Instruct. The learning rate decay adopts a cosine scheduler with a warm-up strategy for the first 3% training steps. The hyperparameter α in Eq. 10 is tuned separately for LLMs due to their different vocabulary size, which is set to 0.1 for CodeLlama and Mistral, and 0.01 for Qwen2.5 and DeepSeek-Math-Instruct. We set $k = 2$ and $\beta = 1.02$ in the floating-point number optimization by default. The ablation for these parameters can be found in **Appendix B**. All the trainings are conducted on 8 NVIDIA A100 GPUs with batchsize 1 per GPU and gradient accumulation step 32, thus resulting in a total batch size of 256.

5.2 EXPERIMENTAL RESULTS

This subsection shows the results of DEL on four PoT code generation settings (GSM8K, MATH, SVAMP, and SimulEq) and three CoT reasoning settings (AQuA, SAT, and MMLU). For other evaluations, we use Qwen2.5-1.5B by default. In **Appendix D**, we provide more exempler visual comparisons for various numerical learning methods.

Benchmark Evaluation. We report the quantitative results in Tab. 2. One can see that DEL produces the best results in most benchmarks and achieves the best average performance across all model architectures, demonstrating consistent improvements over existing methods. It can be seen that the general methods MixCE and EMO can improve mathematical reasoning performance by enhancing logical deduction and language understanding capability. While EMO achieves better overall results than MixCE, it fails in training on CodeLlama and Mistral. This is presumably because that EMO relies on the token embeddings to calculate the similarity between tokens, which may introduce numerical instability in training. The numerical learning methods NTL and DIST^{2}Loss better boost the accuracy since they are designed to improve digit prediction performance. Our DEL better caters to the low-entropy nature of digits, resulting in the best numerical learning performance.

Table 2: Performance comparison of mainstream LLMs trained using their own losses, the general approaches MixCE, EMO, and numerical learning methods NTL, DIST2^{2}Loss and our DEL. Best results are highlighted in **bold**. We report the metrics on average over 3 runs.

Method	Base	mACC	Benchmark						
			GSM8K	MATH	SVAMP	SimulEq	AQuA	SAT	MMLU
Galactica-6.7B (Taylor et al., 2022)	GAL	16.2	10.2	2.2	25.6	4.2	25.6	17.5	28.0
Vicuna-1.5-7B (Zheng et al., 2023)	Llama-2	20.2	13.1	4.9	36.6	4.9	27.2	25.9	30.4
Llama-2-7B (Touvron et al., 2023b)	-	20.5	14.6	2.5	34.5	5.0	30.3	26.8	29.8
CodeLlama-7B (Roziere et al., 2023)	Llama-2	24.3	25.2	13.0	49.4	3.5	24.0	28.6	26.9
MetaMath-7B (Yu et al., 2024)	Mistral	42.2	72.5	29.4	77.5	34.2	24.8	33.2	23.7
DeepMath-Zero-Math-7B (He et al., 2025b)	Qwen2.5-Math	44.6	43.7	50.8	38.7	27.2	44.9	53.6	53.5
Math-Shepherd-RL-7B (Wang et al., 2024a)	Mistral	45.4	82.6	22.6	82.4	45.3	24.4	33.2	27.1
Xwin-Math-V1.1-7B (Li et al., 2024)	Llama 2	45.5	74.7	32.3	81.0	50.6	26.4	27.4	26.0
WizardMath-V1.1-7B (Luo et al., 2023)	Mistral	48.1	74.5	29.3	75.4	70.7	25.3	33.7	28.0
Skywork-OR1-Math-7B (He et al., 2025a)	Qwen2.5-Math	55.1	58.5	80.3	49.6	84.3	50.0	35.0	55.0
MathScale-7B (Tang et al., 2024)	Mistral	61.0	75.6	37.8	80.6	75.7	44.5	62.3	50.6
DART-Math-7B (Tong et al., 2024)	DeepSeek-math	63.1	87.4	52.6	83.8	73.7	38.6	56.8	48.5
			PoT code generation				CoT reasoning		
MLE	CodeLlama	45.0 ±0.30	59.4	33.9	71.2	39.6	38.2	35.0	38.0
MixCE (Zhang et al., 2023)	CodeLlama	45.8 ±0.43	59.7	34.5	72.0	42.5	37.5	33.6	**40.8**
EMO (Ren et al., 2024)	CodeLlama	-	-	-	-	-	-	-	-
NTL-WAS (Zausinger et al., 2025)	CodeLlama	47.1 ±0.24	59.8	35.3	**73.0**	42.8	38.1	42.0	38.9
DIST^{2}Loss (Chung et al., 2026)	CodeLlama	47.4 ±0.24	59.4	34.4	71.7	**45.5**	42.8	38.9	39.2
DEL (Ours)	CodeLlama	**48.6 ±0.30**	**60.0**	**35.8**	**73.0**	43.7	**45.7**	**42.9**	39.3
MLE	Qwen2.5-1.5B	52.2 ±0.49	53.2	42.2	71.9	43.1	45.0	55.0	55.0
MixCE (Zhang et al., 2023)	Qwen2.5-1.5B	52.7 ±0.24	53.5	42.8	72.8	42.9	46.6	54.1	**56.5**
EMO (Ren et al., 2024)	Qwen2.5-1.5B	53.2 ±0.18	54.3	42.5	73.0	46.8	46.8	53.8	55.5
NTL-WAS (Zausinger et al., 2025)	Qwen2.5-1.5B	53.7 ±0.16	54.7	**43.3**	73.3	46.3	43.4	59.1	55.7
DIST^{2}Loss (Chung et al., 2026)	Qwen2.5-1.5B	53.0 ±0.15	54.7	42.1	73.0	44.5	46.5	55.1	55.5
DEL (Ours)	Qwen2.5-1.5B	**55.3 ±0.13**	**56.7**	42.7	**73.5**	**47.8**	**49.3**	**60.9**	55.8
MLE	Mistral	53.9 ±0.41	74.9	39.5	81.3	44.6	48.1	44.5	44.5
MixCE (Zhang et al., 2023)	Mistral	54.7 ±0.47	75.4	39.6	81.0	45.3	48.1	46.8	**46.4**
EMO (Ren et al., 2024)	Mistral	-	-	-	-	-	-	-	-
NTL-WAS (Zausinger et al., 2025)	Mistral	55.4 ±0.18	75.0	39.9	**82.5**	45.3	49.3	49.2	46.2
DIST^{2}Loss (Chung et al., 2026)	Mistral	54.5 ±0.22	74.9	39.9	81.2	43.7	49.9	47.7	44.2
DEL (Ours)	Mistral	**56.3 ±0.21**	**75.7**	**40.2**	82.4	**46.2**	**52.0**	**51.4**	**46.4**
MLE	DeepSeek-math	64.0 ±0.39	76.7	51.1	80.8	53.4	58.2	67.6	60.2
MixCE (Zhang et al., 2023)	DeepSeek-math	64.3 ±0.28	77.0	51.2	80.6	53.9	56.4	69.9	61.2
EMO (Ren et al., 2024)	DeepSeek-math	64.6 ±0.25	77.4	51.0	80.7	**54.4**	57.5	70.0	61.3
NTL-WAS (Zausinger et al., 2025)	DeepSeek-math	64.8 ±0.17	77.0	51.7	81.6	51.6	58.7	72.1	60.9
DIST^{2}Loss (Chung et al., 2026)	DeepSeek-math	64.6 ±0.24	77.0	51.5	81.4	52.6	59.0	69.8	60.9
DEL (Ours)	DeepSeek-math	**66.0 ±0.19**	**77.9**	**51.9**	**83.6**	53.6	**60.3**	**73.2**	**61.4**
MLE	Qwen2.5-7B	67.8 ±0.57	81.2	52.0	86.5	50.6	62.5	76.2	65.8
MixCE (Zhang et al., 2023)	Qwen2.5-7B	68.0 ±0.33	81.6	54.8	86.4	52.5	60.7	73.8	66.3
EMO (Ren et al., 2024)	Qwen2.5-7B	68.1 ±0.32	81.1	55.9	**87.7**	50.9	61.5	74.1	65.7
NTL-WAS (Zausinger et al., 2025)	Qwen2.5-7B	68.6 ±0.37	81.6	54.7	86.2	53.1	60.3	**79.5**	64.5
DIST^{2}Loss (Chung et al., 2026)	Qwen2.5-7B	69.0 ±0.31	82.0	**56.2**	87.2	52.2	63.4	75.4	66.5
DEL (Ours)	Qwen2.5-7B	**70.2 ±0.45**	**81.8**	56.1	87.4	**56.2**	**64.7**	78.0	**67.0**

Numerical answer evaluation. Aside from global task accuracy, we also compare the numerical answers for the methods. We consider two evaluation metric. In Tab. 3a, we calculate the global numerical distance for the predicted numbers, as well as for every digital place. To alleviate the effect of large numbers, we follow (Zausinger et al., 2025) to use the log Mean Absolute Error (MAE) to measure the relative distance between the predicted digits and the ground-truths. We omit reporting very large digital place because the model may predict excessively long digital sequences, and these numbers are rare. In Tab. 3b, we report the numerical accuracy for the predicted numbers, as well as for every digital place. The results shows that our DEL can substantially reduce the numerical distance of the answers even without the explicit injection of numerical distance optimization term, demonstrating a more precise number generation.

Number Uncertainty. To investigate the uncertainty of the predicted numbers, we further statistically analyze the token entropy of the predicted sequence. Based on the full space probability $\mathbf{p}(\cdot|x_{<t})$, we decompose the entropy calculation into two disjoint parts, the digital part and the non-digital part. As shown in Fig. 3, the orginal baseline MLE produces a low digital entropy, approximately 0.1. NTL (Zausinger et al., 2025) significantly reduces digital entropy because its optimization goal is to forcibly push the probability of non-target digits to 0. Another extreme example is DIST^{2}Loss (Chung et al., 2026), which significantly increases the digital entropy, making it even exceed 1. This

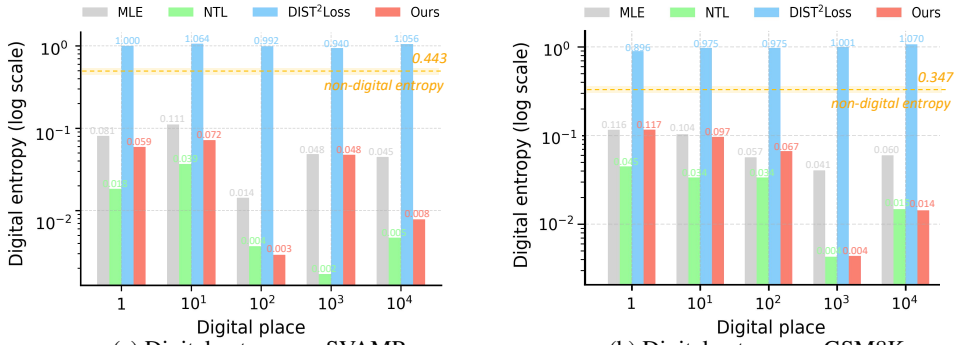
Table 3: Numerical answer evaluation on GSM8K with various loss functions. We report the numerical distance and the task accuracy for the predicted number answers, along with the per digit metric for places, denoted by 1, 10^1, 10^2, 10^3, 10^4, respectively.

(a) Numerical distance evaluation.

Method	MAE per digit					MAE
	1	10^1	10^2	10^3	10^4	
MLE	1.26	1.20	0.91	1.11	1.57	0.72
MixCE	1.25	1.18	0.91	1.08	1.72	0.79
EMO	1.13	1.08	0.85	0.98	1.72	0.71
NTL	**1.16**	1.06	0.84	1.05	1.53	0.78
DIST^{2}Loss	1.18	1.14	0.93	1.11	1.62	0.77
DEL (Ours)	1.17	**1.03**	**0.81**	**0.91**	**1.17**	**0.63**

(b) Numerical accuracy evaluation.

Method	Accuracy per digit					Acc.
	1	10^1	10^2	10^3	10^4	
MLE	66.8	62.4	63.7	66.4	61.7	54.0
MixCE	66.6	62.5	65.6	65.6	57.4	52.8
EMO	69.3	64.1	66.3	66.4	59.6	54.5
NTL	68.9	65.1	64.6	64.9	61.7	55.1
DIST^{2}Loss	67.9	63.5	64.6	61.1	53.2	54.2
DEL (Ours)	**70.0**	**67.0**	**67.2**	64.9	**61.7**	**57.0**



(a) Digital entropy on SVAMP (b) Digital entropy on GSM8K

Figure 3: Comparison of digital entropy between various numerical learning methods.

Table 4: Quantitative comparison between the performance of Qwen2.5 trained using MLE and DEL.

Model	Method	mACC	GSM8K	MATH	SVAMP	SimulEq	AQuA	SAT	MMLU
Qwen2.5-1.5B	MLE	52.8	54.0	42.3	45.3	71.6	44.2	56.4	55.5
	DEL (Ours)	55.4	57.0	42.6	48.8	74.0	48.1	60.9	56.1
Qwen2.5-7B	MLE	68.3	81.4	51.6	63.8	87.0	51.0	76.8	66.3
	DEL (Ours)	70.6	81.7	56.0	65.4	87.6	57.0	78.6	67.9
Qwen2.5-14B	MLE	72.9	87.7	60.9	68.3	88.5	51.2	83.4	70.6
	DEL (Ours)	75.0	88.6	61.1	68.9	90.6	56.2	86.8	72.6

is because DIST^{2}Loss takes the distance-driven soft targets as optimization objectives. It should be noted that the average entropy of non-digit tokens is around $0.3 \sim 0.5$ for all these methods, as depicted by the orange line in Fig. 3. This indicates that DIST^{2}Loss largely rises the uncertainty of number generation, whereas NTL tends to significantly sharpen the probability distribution of digits. The criterion term of DEL makes it a mild method for optimizing number uncertainty.

Evaluation on different model scale. We compare the performance of DEL across 1.5B, 7B and 14B model scales. The results of Tab. 4 show consistent accuracy improvement. It can be seen that computation errors account for a significant proportion and are of considerable importance in mathematical reasoning. This indicates that across the parameter ranges of most widely used open-source models today, digit-level generation uncertainty remains a non-negligible bottleneck, and our DEL brings clear improvements.

6 Conclusion

In this paper, we analyzed the existing numerical learning approaches in large language model training, and found that state-of-the-art methods generally follow the criterion-distance formulation. From the perspective of entropy optimization, we presented digit entropy loss (DEL) to achieve better performance on number generation. We further generalized the numerical learning to floating-point number optimization, enabling the language models with holistic numerical awareness. Experiments on seven mathematical reasoning benchmarks demonstrated the effectiveness of the proposed DEL over prior state-of-the-art numerical learning methods.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in LLM reasoning. In *NeurIPS*, 2025.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3(Feb):1137–1155, 2003.

Jinghao Bian, Mingtao Feng, Weisheng Dong, Fangfang Wu, Jianqiao Luo, Yaonan Wang, and Guangming Shi. Feature information driven position gaussian distribution estimation for tiny object detection. In *CVPR*, pp. 30376–30386, 2025.

Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480, 1992.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, pp. 1877–1901, 2020.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*, 2023.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. InternVL: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *CVPR*, pp. 24185–24198, 2024.

Jiwan Chung, Saejin Kim, Yongrae Jo, Jaewoo Park, Dongjun Min, and Youngjae Yu. Teaching metric distance to autoregressive multimodal foundational models. In *ICLR*, 2026.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pp. 4171–4186, 2019.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In *ICML*, pp. 10764–10799, 2023.

Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *ICML*, pp. 1764–1772, 2014.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *ICML*, pp. 369–376, 2006.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, et al. Skywork open reasoner 1 technical report. *arXiv preprint arXiv:2505.22312*, 2025a.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. DeepMath-103K: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025b.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *ICLR*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *NeurIPS*, 2021b.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *ICLR*, 2020.

Haozhe Ji, Pei Ke, Zhipeng Hu, Rongsheng Zhang, and Minlie Huang. Tailoring language generation models under total variation distance. In *ICLR*, 2023.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. MAWPS: A math word problem repository. In *NAACL*, pp. 1152–1157, 2016.

Chen Li, Weiqi Wang, Jingcheng Hu, Yixuan Wei, Nanning Zheng, Han Hu, Zheng Zhang, and Houwen Peng. Common 7b language models already possess strong math capabilities. *arXiv preprint arXiv:2403.04706*, 2024.

Minghao Li, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. Trocr: Transformer-based optical character recognition with pre-trained models. In *AAAI*, pp. 13094–13102, 2023.

Qiufu Li, Huibin Xiao, and Linlin Shen. BCE vs. CE in deep feature learning. In *ICML*, 2025a.

Ziniu Li, Congliang Chen, Tian Xu, Zeyu Qin, Jiancong Xiao, Zhi-Quan Luo, and Ruoyu Sun. Preserving diversity in supervised fine-tuning of large language models. In *ICLR*, 2025b.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pp. 2980–2988, 2017.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *ACL*, pp. 158–167, 2017.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, pp. 34892–34916, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023.

Raja Marjieh, Veniamin Veselovsky, Thomas L. Griffiths, and Ilia Sucholutsky. What is a number, that a large language model may know it? In *NeurIPS Workshop*, 2025.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Chancharik Mitra, Brandon Huang, Trevor Darrell, and Roei Herzig. Compositional chain-of-thought prompting for large multimodal models. In *CVPR*, pp. 14420–14431, 2024.

Franz Josef Och. Minimum error rate training in statistical machine translation. In *ACL*, pp. 160–167, 2003.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems? In *NAACL*, pp. 2080–2094, 2021.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *ICML*, pp. 28492–28518, 2023.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Siyu Ren, Zhiyong Wu, and Kenny Q. Zhu. EMO: Earth mover distance optimization for auto-regressive language modeling. In *ICLR*, 2024.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. Code Llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *ACL*, pp. 1715–1725, 2016.

Claude Elwood Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

Jiandong Shao, Yao Lu, and Jianfei Yang. Benford's curse: Tracing digit bias to numerical hallucination in LLMs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *ICML*, pp. 31210–31227, 2023.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. Mathscale: Scaling instruction tuning for mathematical reasoning. In *ICML*, pp. 47885–47900, 2024.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*, 2022.

Qwen Team et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. DART-Math: Difficulty-aware rejection tuning for mathematical problem-solving. In *NeurIPS*, volume 37, pp. 7821–7846, 2024.

Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. In *NeurIPS*, volume 37, pp. 34737–34774, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *ACL*, pp. 9426–9439, 2024a.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024b.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. In *NeurIPS*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, pp. 24824–24837, 2022.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *EMNLP*, pp. 38–45, 2020.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*, pp. 11809–11822, 2023.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *ICLR*, 2024.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, and Songfang Huang. How well do large language models perform in arithmetic tasks? *arXiv preprint arXiv:2304.02015*, 2023.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mammoth: Building math generalist models through hybrid instruction tuning. In *ICLR*, 2024.

Jonas Zausinger, Lars Pennig, Anamarija Kozina, Sean Sdahl, Julian Sikora, Adrian Dendorfer, Timofey Kuznetsov, Mohamad Hagog, Nina Wiedemann, Kacper Chlodny, et al. Regress, don't guess–a regression-like loss on number tokens for language models. In *ICML*, 2025.

Shiyue Zhang, Shijie Wu, Ozan Irsoy, Steven Lu, Mohit Bansal, Mark Dredze, and David Rosenberg. MixCE: Training autoregressive language models by mixing forward and reverse cross-entropies. In *ACL*, pp. 9027–9050, 2023.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. In *NeurIPS*, volume 36, pp. 46595–46623, 2023.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. AGIEval: A human-centric benchmark for evaluating foundation models. In *NAACL*, pp. 2299–2314, 2024.

Appendix

In appendix, we provide the following materials:

A. More analysis for various numerical learning methods (referring to Sec. 3 in the main paper).

B. Hyper-parameters study for the loss weight α and the designs in floating-point number optimization (referring to Sec. 5.1 in the main paper).

C. Limitations and future works.

D. Exempler visual comparisons for the numerical learning methods, along with our failure case analysis (referring to Sec. 5.2 in the main paper).

A More Analysis

A.1 Notations

We here give more analysis for the numerical learning. We first lay out some important notations. For simplicity, we consider one multi-digit floating-point number, which is given by Eq. 8 and we write here again:

$$\mathbf{g} = \mathbf{g}_0^L \mathbf{g}_1^{L-1} \cdots \mathbf{g}_L^0 \cdot \mathbf{g}_{L+2}^{-1} \mathbf{g}_{L+3}^{-2} \cdots \mathbf{g}_{L+M+1}^{-M}, \tag{11}$$

where the superscripts and the subscripts of $\mathbf{g}$ represent the digital place and the generation step, respectively. The total sequence U with length T can be represented as

$$U = U_{pre}, \mathbf{x} = U_{pre}, x_0, x_1, \cdots, x_L, \cdots, x_{L+M+1}, \tag{12}$$

where U_{pre} denotes the input sequence preceding the number $\mathbf{x}$. We denote $x_{<t}$ as all preceding input sequences of x_t. The numerical sequence $\mathbf{g}$ can be decoded from the digital token sequences,

$$\mathbf{g} = \text{Decode}(\mathbf{x}). \tag{13}$$

Tab. 5 presents the notations we used in the analysis.

A.2 Analysis to various numerical learning methods

We here give the analysis to various numerical learning methods. We first simplify the symbols by ignoring the token index t since all losses NTL (Zausinger et al., 2025), DIST^{2}Loss and DEL are token-wised and independent of t. We lay out the penalty terms for all the three losses.

$$\begin{aligned}
\mathcal{R}_{NTL} &= \hat{\mathbf{p}}||V_d - \mathbf{g}||_1 = \sum_k \hat{\mathbf{p}}_k d_k, \quad d_k = ||k - \mathbf{g}||_1, \\
\mathcal{R}_{dist} &= \mathcal{L}_{\text{KL}}(\mathbf{q} \parallel \hat{\mathbf{p}}) = -\sum_k \mathbf{q}_k \log \hat{\mathbf{p}}_k - \mathcal{H}(\mathbf{q}), \\
\mathcal{R}_{DEL} &= \hat{\mathbf{p}}\mathbf{BCE}(\dot{\mathbf{p}}, y) = \sum_k \hat{\mathbf{p}}_k B_k = \sum_k \hat{\mathbf{p}}_k \underbrace{\Big(- y_k \log \dot{\mathbf{p}}_k - (1 - y_k) \log \big(1 - \dot{\mathbf{p}}_k\big) \Big)}_{B_k}.
\end{aligned} \tag{14}$$

The partial derivative of the penalty term w.r.t the output logits $\hat{z}_k$ are as follows:

$$\begin{aligned}
\frac{\partial \mathcal{R}_{NTL}}{\partial z_k} &= \hat{\mathbf{p}}_k (d_k - \mathcal{R}_{NTL}), \\
\frac{\partial \mathcal{R}_{dist}}{\partial z_k} &= \hat{\mathbf{p}}_k - \mathbf{q}_k, \\
\frac{\partial \mathcal{R}_{DEL}}{\partial z_k} &= \hat{\mathbf{p}}_k (B_k - \mathcal{R}_{DEL} + \dot{\mathbf{p}}_k - y_k)
\end{aligned} \tag{15}$$

Proposition 1. *The optimal solution of $\mathcal{R}_{DEL}$ is the one-hot label y.*

Proof. The one-hot label y satisfies $y_{\mathbf{g}} = 1$ and $y_k = 0$, $\forall k \neq \mathbf{g}$. Since $\mathbf{BCE}(\dot{\mathbf{p}}, y)$ has the global minima 0 when $\dot{\mathbf{p}} = y$, and by nonnegativity of $\hat{\mathbf{p}}$, one can get $\mathcal{R}_{DEL}$ has the global minima 0, which is also one of the global minima of the Shannon entropy $\mathcal{H}(\hat{\mathbf{p}})$. □

Table 5: Notations.

Notations	Interpretation
$\hat{z} \in \mathbb{R}^{L+M+2,10}$	All the predicted logits for $\mathbf{x}$ on digital tokens.
$\hat{z}_t \in \mathbb{R}^{10}$	The t-th logit vector of x_t, $t \in \{0, 1, \cdots, L+M+1\}$.
$\mathbf{g}_t$	The numerical value of the token x_t.
y_t	The one-hot label of $\mathbf{g}_t$.
$V_d = \{0, 1, \cdots, 9\}$	The digital vocabulary.
$\hat{\mathbf{p}}(\cdot\|x_{<t}) = \mathcal{S}(\hat{z}_t)$	The digital conditional prob. of x_t over V_d, calculated by the softmax function $\mathcal{S}$.
$\hat{\mathbf{p}}(\cdot\|x_{<t}) = \sigma(\hat{z}_t)$	The binary digital conditional prob. of x_t, calculated by the sigmoid function σ.
$\mathbf{q} = \mathcal{S}(-\rho(V_d, \mathbf{g}_t), \tau)$	The soft distribution of numerical distance $\rho(V_d, \mathbf{g}_t)$.
$\mathcal{S}(\cdot, \tau)$	The Softmax function with temperature τ. By default, $\tau = 1$.
$\sigma(\cdot)$	The Sigmoid function.

Table 6: The entropy of the distance distribution $\mathbf{q}$.

The target token $\mathbf{g}$	0, 9	1, 8	2, 7	3, 6	4, 5
$\mathcal{H}(\mathbf{q})$	0.6456	1.0169	1.0708	1.0708	1.0715

Analysis to NTL. We can derive from its partial dericative that NTL encourages to lift up the probabilities for the tokens near the target $\mathbf{g}$. Let $s_k = d_k - \mathcal{R}_{NTL}$, we have the following inequalities,

$$\begin{aligned} s_k = d_k - \sum_{i \neq \mathbf{g}} \hat{\mathbf{p}}_i d_i \leq d_k - \sum_{i \neq \mathbf{g}} \hat{\mathbf{p}}_i = d_k - 1 + \hat{\mathbf{p}}_{\mathbf{g}}, \\ s_k = d_k - \sum_{i \neq \mathbf{g}} \hat{\mathbf{p}}_i d_i \geq d_k - \sum_{i \neq \mathbf{g}} 9\hat{\mathbf{p}}_i = d_k - 9 + 9\hat{\mathbf{p}}_{\mathbf{g}}. \end{aligned} \tag{16}$$

Consequently, we have

$$d_k - 9 + 9\hat{\mathbf{p}}_g \leq s_k \leq d_k - 1 + \hat{\mathbf{p}}_g \tag{17}$$

This indicates that the logits whose tokens is furthest from $\mathbf{g}$ with distance $d_k = 9$ will be definitely reduced since $s_k \geq 9 - 9 + 9\hat{\mathbf{p}}_g > 0$ and thus the partial derivative $\partial \mathcal{R}_{NTL}/\partial z_k > 0$. Likewise, the logits $z_{\mathbf{g}}$ will be increased since $s_k \leq 0 - 1 + \hat{\mathbf{p}}_{\mathbf{g}} = \hat{\mathbf{p}}_{\mathbf{g}} - 1 < 0$ and thus the partial derivative $\partial \mathcal{R}_{NTL}/\partial z_k < 0$. For the logits with medium distance, e.g., $d_k \in [1, 8]$, will be optimized according to $\hat{\mathbf{p}}_g$. It encourages to increase the logits of the token near the target $\mathbf{g}$. When $\hat{\mathbf{p}}_{\mathbf{g}} > 8/9$, indicating the probability of the target token is optimized well, it deduces $s_k > 0$ and all the logits of the non-target tokens will be further reduced. To sum up, NTL gradually shifts the probability mass from tokens far from $\mathbf{g}$ to the tokens near $\mathbf{g}$, and finally $\mathbf{g}$ alone.

Analysis to DIST^{2}Loss. By the property of KL divergence, the optimal solution of $\hat{\mathbf{p}}$ equals to the soft target $\mathbf{q}$. In this case, one can obtain the digital conditional entropy,

$$\mathcal{H}(\hat{\mathbf{p}}) = \mathcal{H}(\mathbf{q}) = -\sum_k \mathbf{q}_k \log \mathbf{q}_k \tag{18}$$

Since $\mathcal{H}(\mathbf{q})$ is a constant, it can be directly computed. There is totally 10 possible ground-truth digits for $\mathbf{g}$. Due to the symmetry of $\mathbf{q}$, it can be reduced to 5 cases. We list the computation results of $\mathcal{H}(\mathbf{q})$ in Tab. 6. One can see that $0.6456 \leq \mathcal{H}(\mathbf{q}) \leq 1.0715$, which is much larger than the token entropy as we have shown in Fig. 3. While the cross-entropy term $\mathcal{H}(\mathbf{p}, \mathbf{g})$ attempts to optimize $\mathbf{p}$ to an one-hot distribution, the penalty term of DIST^{2}Loss still enlarges the digital entropy because the target itself has a large entropy. This is why we observe the phenomenon of increasing digital entropy from Fig. 3 that the digital entropy of DIST^{2}Loss is approximately 1.0.

Analysis to Digit Entropy Optimization.

Digit entropy optimization is not new. For example, it has been independently discussed in (Bian et al., 2025) for object detection and in (Agarwal et al., 2025) for LLM reasoning, where entropy minimization is observed to be useful. Others take the opposite approach: maximum entropy principle (Li et al., 2025b; Wang et al., 2025) to encourage the models to evolve more diversity of

Table 7: Quantitative comparison between digital focal loss and our DEL. The base model is Qwen2.5-1.5B.

Method	mACC	GSM8K	MATH	SVAMP	SimulEq	AQuA	SAT	MMLU
MLE	52.8	54.0	42.3	71.6	44.2	45.3	56.4	55.5
Digital Focal Loss	53.0	55.9	43.0	72.8	44.7	45.3	54.5	54.8
NTL-WAS (Zausinger et al., 2025)	53.8	55.1	**43.5**	73.3	46.1	43.1	59.1	56.3
DIST^{2}Loss (Chung et al., 2026)	53.1	54.2	41.9	73.2	44.6	45.7	55.9	56.5
Digit Entropy Optim. (Eq. 6)	54.2	54.6	42.4	72.7	47.7	47.2	60.5	54.2
DEL (Ours)	**55.4**	**57.0**	42.6	**74.0**	**48.1**	**48.8**	**60.9**	**56.1**

thought chains. To chase for deterministic number generation, we consider entropy minimization, as presented in Eq. 6. In the 5-th row of Tab. 7, one can see that the overall performance of digit entropy optimization already surpasses previous numerical learning methods, e.g., NTL and DIST^{2}Loss. Due to its unsupervised nature, it is free from the issue of numerical distance. However, this entropy loss has many trivial solutions. Any one-hot distribution can reach the global minima. Our DEL leverages the digit conditional probability and the binary cross-entropy, enabling a supervised entropy optimization manner. The last row of Tab. 7 show that our DEL achieves the best mACC on the 7 benchmarks, demonstrating the effecttiveness of DEL.

Analysis to Focal Loss. Adopting binary cross-entropy loss in multi-class learning is not new either. It has been recently demonstrated in (Li et al., 2025a) that BCE can maximize the intra-class compactness and inter-class distinctiveness among sample features, which makes it has potential to achieve better classification performance than standard cross-entropy loss. Early research focal loss (Lin et al., 2017) also adopted BCE as the core objectives in object detection. It was originally proposed for addressing extreme class imbalance problem in object detection, where too many negative anchor boxes dominant the gradient flow and thus does harm to detection performance. We directly integrate the application of focal loss in numerical learning without any adaptation changes from object detection to language model numerical learning. Under our defined symbol system, integrating focal loss into numerical learning can be written as

$$\mathcal{L}_{FL} = \alpha \sum_k A_k^\gamma B_k = \alpha \sum_k \Big(y_k(1-\dot{\mathbf{p}}) + (1-y_k)\dot{\mathbf{p}}\Big)^\gamma \Big(-y_k \log \dot{\mathbf{p}}_k - (1-y_k)\log\big(1-\dot{\mathbf{p}}_k\big)\Big) \quad (19)$$

It can be seen that focal loss models each token as a binary classification task on $|V|$ classes. Without the softmax-based probabilities, it is difficult to modeling the inner relationship of digits and is inconsistent with the inference phase. In Tab. 7, we compare our DEL with the digital focal loss. The experimental results demonstrate that DEL can achieve better accuracy on 6 of 7 benchmarks.

B Hyper-parameters study

Loss weight α in DEL. We investigate the impact of loss function weights on model performance. Tab. 8a reports the quantitative results. Due to the varying vocabulary sizes of different LLMs, such as 151,936 for Qwen 2.5, 102,400 for DeepSeek-Math-Instruct, 32,016 for CodeLlama, and 32,000 for Mistral, the setting of α will be adjusted accordingly. We set $\alpha = 0.01$ for Qwen2.5 and DeepSeek-Math-Instruct and $\alpha = 0.1$ for CodeLlama and Mistral in other experiments by default.

k and β in floating-point number optimization. In Eq. 9, we assign a linear penalty for integers and an exponential decay for decimals. According to the results in Tab. 8b, we set $k = 2$ by default in all the other experiments. As for the decimals, we conduct various options. Tab. 8c shows us interesting results that the performance slightly drops if we set no penalty or a constant penalty for the decimals. What if we treat the decimal as a new integer (as done by DIST^{2}Loss, see Fig. 4), the performance will degrade significantly. This indicates that the discontinuous penalty between integers and decimals leads to suboptimal results. In contrast, our floating-point number optimization can better match the continuity of numerical values, further boosting the performance gains. We set $\beta = 1.02$ by default.

Table 8: **Ablations**. We show ablation experiments for DEL.

(a) Loss weight α in DEL.

α	Accuracy	MAE
0.02	56.0	0.69
0.01	**57.2**	**0.65**
0.005	56.8	0.66
0	54.0	0.71

(b) Linear weighting k in integers in floating-point number optimization.

k	Accuracy	MAE
0.5	56.5	0.68
1	56.8	0.68
2	**57.2**	**0.65**
5	56.2	0.72

(c) Exponential decay β in decimals in floating-point number optimization.

β	Accuracy	MAE
∞	56.4	0.69
1	56.6	0.66
1.02	**57.2**	**0.65**
as integers	55.2	0.74

Table 9: Evaluation of diverged components in numerical learning.

Digit probability $\hat{\mathbf{p}}(\cdot\|x_{<t})$	DEL in Eq. 7	Distance term in NTL	Floating-point number optim.	mACC
				52.9
✓		✓		53.8
✓		✓	✓	54.4
	✓			55.1
	✓	✓		54.3
	✓		✓	55.4

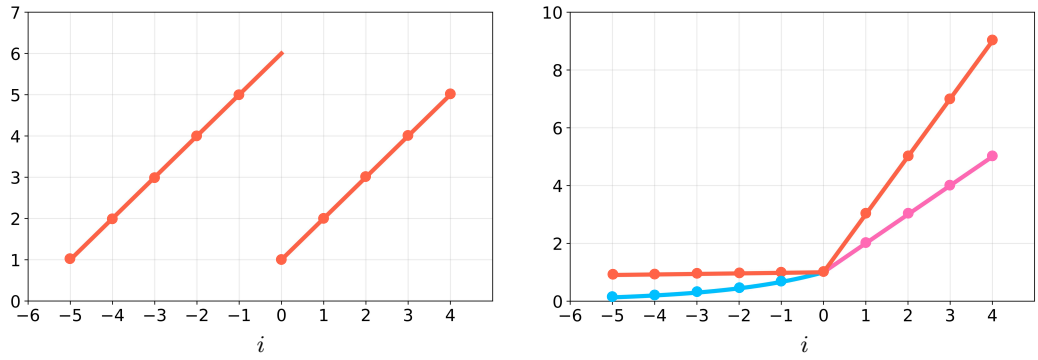
Evaluation of diverse components in numerical learning. We conduct experiments to check out the effect of each component in numerical learning. As shown in Tab. 9, we start from MLE, a clean baseline of 52.9 mACC on seven benchmarks. The second row is NTL (Zausinger et al., 2025), which adopts the digital probability $\hat{\mathbf{p}}(\cdot|x_{<t})$ as the criterion term, along with the numerical distance, achieving a mACC of 53.8. It can be further improved to 54.4 if we intergrate our floating-point number generation into NTL, as shown by the third row. Then, in 4-th row, our DEL without any other components produces 55.1 mACC, demonstrating the superiority of the new criterion term. If we add the distance term of NTL, it degrades to 54.3, as shown in the results of 5-th row. As we have analyzed in Sec. 4, the numerical distance terms injects a strong geometric prior into the LLMs, which has a deviation from the semantic information of digits. Thus we deprecate the distance term in our formulation. Finally, combined with the floating-point optimization, the performance can be further improved to 55.4 mACC (see the last row). This gives us our final DEL formula (Eq. 10).

C Limitations and Future Works

While demonstrating consistent improvements over previous methods, DEL has some limitations. Firstly, it is still optimized in a token-level manner, without joint modeling of the global numerical semantics of the number sequence. Secondly, DEL currently extends numerical learning to floating-point number optimization, but has not yet expanded many mathematical operations. Thirdly, we acknowledge that reasoning quality of LLM is the result of the combined effect of multifaceted factors. If we want to improve mathematical reasoning capabilities, this may require taking a big step forward, including improving the reasoning logic, such as using better CoT hints, and improving the model's understanding of mathematical operators. At the same time, it also has some other problems to solve, such as sub-task decomposition, induction and summarization, and rule application, which makes improving LLM mathematical reasoning ability very challenging. The line of works in numerical learning show that digit generation is a non-trivial failure mode in current math reasoning. In the future, one may consider modeling mathematical operations in LLMs, such as addition, multiplication, division, exponentiation, etc., enabling LLMs with more numerical reasoning capabilities.

D Visualization

We provide several visual comparison for the numerical learning methods. We highlight the reasoning errors in **red** bold. We use Qwen2.5-1.5B to inference. In Fig. 5, Fig. 6, Fig. 7, Fig. 8, we show that accurate number prediction is crucial to mathematical reasoning as a small error in number prediction would lead to a complete wrong answer to the mathematical problem.



(a) $u(t)$ in DIST^{2}Loss (b) $u(t)$ in our method

Figure 4: Illustration of floating-point optimization. Our DEL adopts linear penalty for integers ($i \geq 0$) and exponential decay in decimals ($i < 0$), while previous work DIST^{2}Loss has a discontinuous penalty. Different colors represent different parameter choices in $u(t)$.

We also present some failure cases that the current numerical learning is hard to handle. The first one is irrelevant context distraction (Shi et al., 2023), as shown in Fig. 9. This error is irrelevant to number prediction accuracy, which is simply caused by the distracted attention to the irrelevant contexts. The second error is calculation error. The reasoning is correct, however, the calculation results are wrong. As we have concluded in Sec. 3, numerical reasoning requires language models to possess a range of capabilities, such as logical reasoning, symbolic comprehension, unit conversion, problem decomposition, contextual semantic association. We believe that in the future, explicitly injecting more numerical capabilities into the optimization process of language models will help improve numerical reasoning.

AI USE STATEMENT

In this work, we used generative AI tools for polish writing and searching related works. Additionally, we used generative AI tools for analyzing the codes. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work has no ethic issue.

REPRODUCIBILITY STATEMENT

The source codes and the trained models of this work will be publicly available.

Question:

$$3x + x + x + x - 3 - 2 = 7 + x + x$$

In the equation above, what is the value of x ?
Answer Choices: (A) $-\frac{5}{7}$ (B) 1 (C) $\frac{12}{7}$ (D) 3
Correct: D

MLE:
We can combine like terms on the left side of the equation to get

$$3x + x + x + x - 3 - 2 = 7 + x + x$$

$$\mathbf{4}x - 5 = 7 + 2x$$

We can subtract $2x$ from both sides to get

$$2x - 5 = 7$$

Then we can add 5 to both sides to get

$$2x = 12$$

Finally, we can divide both sides by 2 to get

$$x = 6$$

The answer is B

NTL:
3x+x+x+x-3-2=7+x+x
3x+x+x+x-5=7+2x
3x+x+x+x-2x=7+5
3x=12
x=12/3
x=4
The answer is B

DIST^{2}Loss:

$$3x + x + x + x - 3 - 2 = 7 + x + x$$

$$\mathbf{4}x - 5 = 7 + 2x$$

$$2x = 12$$

$$x = 6$$

The answer is D

DEL (Ours):
3x+x+x+x-3-2=7+x+x
3x+x+x+x-5=7+2x
3x+x+x+x-2x=7+5
3x+x=12
4x=12
x=3
The answer is D

Figure 5: Visualization of prediction results of LLM. The reasoning errors are highlighted in **red** bold.

Question: John plans to sell all his toys and use the money to buy video games. He has 13 lego sets and he sells them for $15 each. He ends up buying 8 video games for $20 each and has $5 left. How many lego sets does he still have?
Correct: 2

MLE:
He earned 13*15=195 dollars from selling his lego sets
He spent 8*20=160 dollars on video games
He has 195-160-5=30 dollars left
He originally had 30+160=190 dollars
He has 190/15=12.666667 lego sets left
The answer is 12

NTL:
He earned 13*15=195 dollars
He spent 8*20=160 dollars on video games
He has 195-160-5=30 dollars left
He has 195-30=165 dollars left to buy lego sets
He has 165/15=11 lego sets left
The answer is 11

DIST^{2}Loss:
He earned 13*15=195 dollars from selling his lego sets
He spent 8*20=160 dollars on video games
He has 195-160-5=30 dollars left
He had 195-30=165 **lego sets**
He sold 195-160=35 lego sets
He has 165-35=130 lego sets left
The answer is 130

DEL (Ours):
He sold 13 lego sets for 13*15=195 dollars
He spent 8*20=160 dollars on video games
He had 195-160-5=30 dollars left
He had 195-30=165 dollars from selling legos
He had 13-165/15=13-11=2 legos left
The answer is 2

Figure 6: Visualization of prediction results of LLM. The reasoning errors are highlighted in **red** bold.

Question: Charlotte went into the kitchen supply store knowing she wanted a set of pot and pans for $120.00, a set of mixing bowls for $20.00 and 5 separate utensils at $5.00 each. At check out, the clerk told her everything was 20% off. How much was her order?
Correct: 132

MLE:
She had a $120.00 set of pots and pans and a $20.00 set of mixing bowls so that's 120+20 = $140.00
She had 5 separate utensils at $5.00 each so that's 5*5 = $25.00
Her order was $**140.00** for pots and pans, $20.00 for mixing bowls and $25.00 for utensils for a total of 140+20+25 = $185.00
Her order was $185.00 and it was 20% off so that's .20*185 = $37.00
Her order was $185.00 and it was 20% off so she paid $37.00 less so her order was 185-37 = $148.00
The answer is 148

NTL:
She had a $20.00 mixing bowl and 5 separate utensils at $5.00 each so that's 20+5*5 = $**35.00**
Her order was $120.00 for the pots and pans, $20.00 for the mixing bowls and $35.00 for the utensils for a total of 120+20+35 = $175.00
Her order was $175.00 and it was 20% off so that's 175*.20 = $35.00
Her order was $175.00 and it was 20% off so that's 175-35 = $140.00
The answer is 140

DIST^{2}Loss:
She had a $120.00 set of pots and pans and a $20.00 set of mixing bowls so that's 120+20 = $140.00
She had 5 separate utensils at $5.00 each so that's 5*5 = $25.00
Her order was $**140.00** for pots and pans, $20.00 for mixing bowls and $25.00 for utensils for a total of 140+20+25 = $185.00
Her order was $185.00 and it was 20% off so that's 185*.20 = $37.00
Her order was $185.00 and it was 20% off so that's $185-$37 = $148.00
The answer is 148

DEL (Ours):
She had a set of pots and pans for $120.00 and mixing bowls for $20.00 so her order was $120.00 + $20.00 = $140.00
She had 5 separate utensils at $5.00 each so that was 5 * $5.00 = $25.00
Her order was $140.00 + $25.00 = $165.00
Her order was $165.00 and it was 20% off so that's 20/100 * $165.00 = $33.00 off
Her order was $165.00 - $33.00 = $132.00
The answer is 132

Figure 7: Visualization of prediction results of LLM. The reasoning errors are highlighted in **red** bold.

Question: Valerie earns $5000 per month, 1/2 of what her brother earns. If their mother earns twice their combined salary, what's the total amount of money they all have together?
Correct: 45000

MLE:
Valerie earns **$5000/2 = $2500** per month.
Valerie's brother earns $2500/2 = $1250 per month.
Their combined salary is $2500 + $1250 = $3750 per month.
Their mother earns 2 * $3750 = $7500 per month.
The total amount of money they all have together is $5000 + $3750 + $7500 = $16250.
The answer is 16250

NTL:
Valerie earns 1/2 of what her brother earns, meaning her brother earns 5000*2=10000 per month.
Together, Valerie and her brother earn 10000+5000=15000 per month.
Their mother earns twice their combined salary, meaning she earns 2*15000=30000 per month.
The total amount of money they all have together is 30000+15000=45000 per month.
The answer is 45000

DIST^{2}Loss:
Valerie earns **$5000/2 = $2500** per month.
Her brother earns $2500 * 2 = $5000 per month.
Their combined salary is $2500 + $5000 = $7500 per month.
Their mother earns 2 * $7500 = $15000 per month.
The total amount of money they all have together is $15000 + $7500 = $22500.
The answer is 22500

DEL (Ours):
Valerie's brother earns 2 * 5000 = 10000 dollars per month.
Valerie's mother earns 2 * (5000 + 10000) = 30000 dollars per month.
The total amount of money they all have together is 5000 + 10000 + 30000 = 45000 dollars.
The answer is 45000

Figure 8: Visualization of prediction results of LLM. The reasoning errors are highlighted in **red** bold.

Irrelevant context distraction

Question:During summer break 644997 kids from Lawrence county stayed home and the other 893835 kids went to camp. An additional 78 kids from outside the county attended the camp. How many kids are in Lawrence county?
Correct: 1538832
Answer: First add the number of kids from Lawrence county who stayed home to the number of kids who went to camp: 644997 + 893835 = 1538832
Then add the number of kids from outside the county who attended the camp to find the total number of kids in Lawrence county: 1538832 + 78 = 1539010
The answer is 1539010

Calculation error

Question: Question: Michael rented a truck for one day. There was a rental fee of 20.99 dollars , and there also was a charge of 25 cents for each mile. Michael paid 95.74 dollars when he returned the truck. For how many miles did he drive the truck?
Correct: 299
Answer: Let x be the number of miles Michael drove the truck.
The total cost is 20.99 + 0.25*x = 95.74
Subtracting 20.99 from both sides, we get 0.25*x = 74.75
Dividing both sides by 0.25, we get x = **303**
The answer is 303

Figure 9: Failure cases. The reasoning errors are highlighted in **red** bold.