arXiv:2601.12095v1 [cs.LG] 17 Jan 2026

神经同构域：基于 Transformer 的代数数值嵌入

HamidrezaSadegh Saeedeh Momtazi Reza Safabakhsh
阿米尔卡比尔理工大学计算机工程系，德黑兰，
伊朗
{sadeghihamid, momtazi, safa}@aut.ac.ir

2026年1月21日

摘要

神经网络模型在处理极小或极大的数字时，往往会遇到挑战，原因包括上溢、下溢和输出波动不稳定等问题。为缓解这些问题，我们提出使用数字的嵌入向量，而非直接使用其原始值。这些嵌入旨在保留关键的代数性质，同时避免数值不稳定性。本文首次提出一种定长数字嵌入向量，可在有理数域中保留代数运算——加法、乘法和比较。我们提出了一种新颖的神经同构域，它是对群和域等代数结构的神经抽象。这个神经域的元素是嵌入向量，在计算过程中能够保持代数结构。我们的实验表明，加法表现出色，在单位元、封闭性和结合律等关键代数测试中，准确率超过**95%**。相比之下，乘法则面临挑战，在各项代数性质上的准确率介于**53% 至 73%**之间。这些发现凸显了模型在通过加法保留代数性质方面的优势，同时也指出了进一步改进乘法处理能力的方向。

1 引言

数值数据的表示与处理是人工智能诸多领域中的核心挑战，包括自然语言处理（NLP）和计算机视觉（CV）。表示和处理数字的一种方法是将其嵌入。数字嵌入与数码嵌入之间存在一个关键区别：前者将数字视为离散的语言词元（例如，将 25 视为一个单词） [1], ，而后者旨在编码数值的连续定量属性（例如，实数 25） [2, 3]。现有的大多数研究主要关注数字嵌入，并致力于解决数字词元的词汇变化性和语境用法等挑战 [1, 4]。然而，这些方法往往无法体现数字所固有的深层代数结构。为此，我们提出了一种名为神经同构域 （*NIF*）的新框架。该框架利用代数原理构建数字嵌入，以保留有理数的内在数学性质，从而实现更丰富且语义更明确的表示。

深度学习的进展表明，嵌入空间能够捕捉数据中的复杂关系。例如，静态词嵌入，如 Word2Vec [5], GloVe [6] 和 FastText [7] ，以及上下文化词嵌入，如 BERT [8], XLNet [9], BART [10], ELECTRA [11], ALBERT [12], ERNIE [13], T5 [14], BigBird [15] 和 DeBERTa[16] ，都展示了向量空间在保留词语间句法和语义关系方面的强大能力。同样，图嵌入 [17, 18, 19, 20, 21] 在保留图的结构性质方面也取得了成功。这些方法启发我们探索嵌入空间在保留数字代数性质方面的潜力。
在数值数据领域，神经算术逻辑单元（NALU） [22] 和神经算术单元（NAU） [23]等研究提出了专为执行算术运算而设计的神经网络架构。这些架构能够显式地学习加法和乘法运算，标志着将算术能力整合到神经网络中取得了重大进展。此外，神经图形处理器 [24] 等方法

以及程序执行网络 [25] 展示了神经网络执行算法（包括算术运算算法）的潜力。由于这些模型不提供数字嵌入，因此无法应用于实际场景。Jiang 等人 [1] 提出了两种新颖的数字嵌入方法，以解决数字的未登录词（OOV）问题。然而，这些嵌入并不完全具备理解和处理数字或数的能力，这种能力称为 数值能力 [26]。数值能力包括理解数字或数的顺序、大小、单位及其算术运算。本文提出的 NIF 嵌入为每个数生成一个独特的嵌入向量，并通过保留结构和代数性质，满足数值能力的要求。

为了保持代数性质，我们构造数字嵌入，使所得嵌入向量集合紧密近似于有理数的同构结构。在加法和乘法运算下，有理数和实数构成有序域 [27, 28]。鉴于有理数是可数的 [29], ，我们可以在其上定义一个概率分布，使分配给任意单个数的概率均非零。该集合构成了我们框架中同构映射的基础。

需要强调的是，尽管理论构造假设在形成同构域时满足理想条件，但神经网络本身会引入误差，因此实践中无法实现这些理想条件。因此，嵌入向量上所诱导的结构只能近似于到有理数域的同构映射，而不能完美实现。

为了有效嵌入数字，首先必须确定它们的表示方法。在 NALU 和 NAU 等先前模型中，数字采用浮点数或整数格式表示。此外，一些早期方法采用了科学记数法 [30] 或子词分词方法。字节对编码（BPE） [31], 、WordPiece [32], 和 SentencePiece [33] 等分词技术广泛用于语言模型（LM） [34, 35]和大语言模型（LLM） [36, 37, 38] ，以表示单词和数字。然而，我们的模型需要全面而精确地表示数字，尤其是因为神经网络会因下溢或上溢问题而无法处理非常大或非常小的数字。因此，我们不依赖科学记数法或分词等传统方法，而是将数字表示为由各个位数组成的序列。这种方法确保神经网络能够有效处理所有数字，而不会遇到数值限制。

鉴于我们提出的表示方法将每个数编码为数字序列，循环神经网络（RNN）或 Transformer 架构 [39] 最适合用于嵌入此类序列。对于特别长的序列，Sparse Transformer [40] 和 LongFormer [41], 等 Transformer 模型的专用变体旨在高效处理较长序列。然而，即使是 20 位的非负整数，在实际嵌入中也被视为非常大的数。因此，我们采用标准 Transformer 作为模型，因为它能够充分处理这一长度的序列，无需使用更专门的替代方案。

本文的主要贡献如下：

- 据我们所知，首次提出 NIF 框架，在保留数字代数性质的同时对其进行嵌入。
- 确立理想条件，并证明定理和引理，从而构造出一个与有理数域同构的有序域。
- 提出一套全面的评估标准，用于评估加法和乘法运算符。

本文结构安排如下：第 2 节回顾数字嵌入的既有方法及其在保留加法和乘法等代数性质方面的局限性。第 3 节介绍我们提出的新方法，用于构造能够保持这些性质的数字嵌入。第 4 节定义评估指标，以评估单位元、封闭性和可逆性等代数性质。第 5 节详细介绍实验设置、数据生成、模型架构和结果，并重点关注算术运算。最后，第 6 节总结研究发现，第 7 节讨论未来方向。

2 相关工作

近年来，神经网络中的数值输入处理受到了研究人员的广泛关注。我们的工作在这些研究成果的基础上进行了拓展，提出了一种能够保留代数性质的新型数字嵌入方法。该领域的相关工作可分为四个主要方向：数字嵌入（2.1）、数码嵌入（2.2）、神经网络中的算术运算（2.3）以及神经网络的代数方法（2.4）。本节将回顾这四个领域的相关工作。

2.1 数字嵌入

近年来，数字嵌入领域的进展提出了多种方法，以改进语言模型中数值的表示。Zhou等人 [2] 提出了*FoNE*，这是一种采用傅里叶特征对数字进行编码的单词元嵌入方法。通过基于数字位数应用傅里叶变换，FoNE 构建了紧凑且精确的嵌入。给定长度为 d 的嵌入向量，该方法最多可以嵌入具有 $d/2$ 位数的数字。需要注意的是，FoNE 的目标并不是在嵌入空间中实现直接的算术运算，例如加法或乘法（即，它不会在嵌入空间中构建一个域）；相反，它通过在大语言模型中应用数字嵌入来评估其执行此类运算的有效性。

作为一种互补方向，Golkar等人 [3] 提出了*xVal*，这是一种专为科学语言模型设计的连续嵌入方法。xVal 将数字表示为连续向量，同时保留大小和相对差异等关键数值属性，从而提升模型在物理学和工程学等领域中的性能；在这些领域，离散化可能会带来显著限制。

Han等人 [42] 通过*LUNA*为这一研究方向作出了贡献。LUNA是一种旨在提升Transformer对数字理解能力的框架。LUNA引入了模块化的“数字插件”架构，通过专门的增强机制区别对待数字和文本词元。这一机制配备了专用嵌入层，能够更有效地捕捉数字的数值和结构格式，从而丰富数字表示。

在基于数字的编码方案基础上，Sivakumar和Moosavi [43] 提出了一种数字嵌入的显式聚合方法。他们的方法将数字分解为独立的位数，为每个位数生成嵌入，然后根据反映十进制数制的位置权重计算这些嵌入的加权和。随后，这一聚合嵌入通过在输入序列中引入特殊词元，或在训练过程中加入辅助损失函数，整合到Transformer模型中。不同于以往依赖模型隐式学习数字嵌入聚合的方法，这种显式方法提供了直接监督，增强了模型的数值推理能力，同时无需进行架构修改。

总体而言，所有这些方法的目标都是将数字映射为定长向量。此外，除 xVal 外，所有方法都依赖基于数字的编码方案，因此能够稳健地处理非常大或非常小的数字。在这方面，它们与我们工作所采用的方法非常接近。

2.2 数码嵌入

Jiang 等人提出了两种数字嵌入方法，用于处理未登录数字 [1]。他们的方法采用自组织映射（SOM）或高斯混合模型（GMM），生成有限个原型数码，并将数字的嵌入表示为这些原型的加权平均值。尽管该方法对某些任务有效，但它并未明确保留代数运算。

Sundararaman等人 [4] 为数码提出了 DICE（确定性、独立于语料库的嵌入），其余弦相似度反映实数上的实际距离。他们还提出了一种正则化方法，用于学习上下文中的基于模型的数字嵌入。

2.3 神经网络中的算术运算

Franco 和 Cannas [44] 设计了能够执行基本算术运算的前馈多层神经网络。然而，他们的工作侧重于解决特定任务，而非将算术功能内在地嵌入数字表示中。

随后，Trask 等人 [22] 提出了神经算术逻辑单元（NALU），通过明确的机制增强神经网络，使其能够执行算术运算。尽管 NALU 能够对两个输入执行加法和乘法等运算，但它并未为单个数字提供专门的嵌入。

最近，McLeish 等人 [45] 证明，配备了适当结构化嵌入的标准 Transformer 架构能够执行算术任务。他们提出了**算盘嵌入**这一方法，通过编码每个数字相对于数的起始位置，使 Transformer 无需进行任何架构修改，便能在算术基准测试中达到最先进的性能。

2.4 神经网络的代数方法

Parada-Mayorga 和 Ribeiro [46] 研究了基于交换代数的代数神经网络（AlgNNs），通过代数信号处理将多种架构统一起来。他们关于形变稳定性的研究，为理解神经网络的代数性质提供了深入见解，而我们的研究正是在此基础上展开的。

Zimmer [47] 通过定义一种新的序贯复合概念，朝着建立神经网络代数理论迈出了步伐。这项工作聚焦于网络复合，并未将代数概念直接应用于数的表示。

3 所提方法

我们的目标是提出一个与有理数域同构的有序域。我们假设读者熟悉群、域、同构和序的定义[1]。

对该域的介绍促成了为每个有理数构造定长嵌入向量，该向量能够保留代数运算；同时也展示了代数性质的神经网络版本，并引入了神经同构域这一概念。这种新颖的方法旨在解决神经网络输入中处理极小数、小数、大数和极大数所面临的挑战，同时保留关键的数学性质。

3.1 数的表示

要嵌入数字，必须确定其表示格式。一种表示大数和小数的实用方法是使用由数字和符号（正号、负号、小数点）组成的序列。因此，数$\mathbf{x}$表示为序列 $\{x_i\}_{i=1}^{n}$，其中 $x_i \in \{0, \ldots, 9, \cdot, -, +, s\}$（其他符号表示也请参见表1）。要在解码器中重构该数，需要一个起始词元（s），因此模型接收的输入为14个词元。由于已知该数的长度，因此不需要结束词元。

符号表示	缩写	描述
$\mathbb{Q}$	-	有理数集
$\mathbb{R}$	-	实数集
φ_{Θ_E}	φ	函数 φ_{Θ_E}：$\mathbb{Q} \to \mathbb{R}^d$ 等价于带有可训练参数 Θ_E的编码器神经网络
$\varphi^{\dagger}_{\Theta_D}$	$\varphi^{\dagger}$	函数 $\varphi^{\dagger}_{\Theta_D}$：$\mathbb{R}^d \to \mathbb{Q}$ 等价于带有可训练参数 Θ_D的解码器神经网络
$*$	-	不带任何可训练参数的自定义二元运算符（$\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$）
$*_\Theta$	$\dot{*}$	带有可训练参数 Θ的自定义二元运算符（$\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$）
$+$	-	定义在有理数和实数上的通常加法运算符
$+_i$	-	不带任何可训练参数的自定义二元加法运算符
$+_\Theta$	$\dot{+}$	带有可训练参数 Θ 的自定义二元加法运算符
$\times$	-	定义在有理数和实数上的常规乘法运算符
$\times_i$	-	不带任何可训练参数的自定义二元乘法算子
$\times_\Theta$	$\dot{\times}$	带有可训练参数 Θ 的自定义二元乘法算子
ω_Θ	ω	函数 ω_Θ：$\mathbb{R}^d \times \mathbb{R}^d \to [0,1]^3$ 与带有可训练参数 Θ 的神经序等价
$<_{\omega_\Theta}$	$<_\omega$	使用 ω_Θ 构造的序
$<$	-	有理数和实数上的通常序
$<_i$	-	不含任何可训练参数的序
D	-	概率质量函数为 f 的离散分布，使得对于所有 $a \in \mathbb{Q}$，$f(a) > 0$
H	-	$H = \varphi(\mathbb{Q})$
h 或 h_i	-	H的每个元素
$Img(f)$	-	函数 f 的像
$d(.,.)$	-	度量 d：$\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$。它可以被视为一个损失函数
$(F, +_i, \times_i, <_i)$	$(F, +_i, \times_i)$	域 F，其加法运算符为 $+_i$，乘法运算符为 $_i$，序为 $<_i$
$(F, \dot{+}, \dot{\times}, <_\omega)$	$(F, \dot{+}, \dot{\times})$	域 F，其加法运算符为 $\dot{+}$，乘法运算符为 $\dot{\times}$，序为 $<_\omega$
$\mathbf{x}$	-	将数表示为数字序列（$\mathbf{x} = \{x_i\}_{i=1}^{n}$）
a, b	-	实数和有理数的标准表示
$1_{\{true\}}(\circ)$	$1(.)$	指示函数

表1：本文的符号表

3.2 模型架构

NIF 模型与 BART 模型类似 [10], ，结合了双向 Transformer 和自回归 Transformer，构成一个自编码神经网络。不过，我们的模型有两个关键区别：（1）它不进行去噪；（2）它输出的不是一个矩阵，而是来自 Transformer 编码器的单个向量，然后将该向量传递给解码器。这个向量用作输入序列的嵌入。因此，如果$\mathbf{x} = \{x_i\}_{i=1}^{n}$ 是编码器输入数，则$\mathbf{x}$ 的嵌入向量 h 将被生成。

[1]为简洁起见，本文不提供这些定义；读者可参阅抽象代数书籍 [27], 、数学分析 [28], 和集合论 [29] ，以获取详细说明。

Transformer架构的一项关键优势是，它能够将行数可变的矩阵作为隐藏状态从编码器传递到解码器，这一能力使其有别于传统的RNN编码器-解码器模型。然而，在我们提出的模型中，每个编码器输入都需要一个唯一的向量，因此我们对标准Transformer架构进行了调整，以满足这一特定要求。

由于 Transformer 解码器需要矩阵作为其隐藏输入，因此可以将所得向量作为单行矩阵提供给解码器。然而，这种方法无法向解码器明确提供有关该数长度的信息。另一种方法是将向量 $n+1$ 重复多次，构造一个 $(n+1)$ 行矩阵。鉴于该矩阵的秩为 1，我们在每一行中加入位置编码向量 $\{p_1, \ldots, p_{n+1}\}$，将矩阵的秩提升至$n+1$。实验结果表明，后一种方法优于前一种方法。相应的模型架构如图1所示。

我们将编码器函数记为 φ，将解码器函数记为 $\varphi^{\dagger}$。目标是构造集合 $H = \varphi(\mathbb{Q})$，并定义运算 $+_{\Theta_1}$ 和 $\times_{\Theta_2}$，使得元组 $(H, +_{\Theta_1}, \times_{\Theta_2})$ 构成一个有序域（$\mathbb{Q}$ 是所有有理数的集合）。为使这一构造成立，函数 φ 必须是一个同构。第3.3节探讨这些运算的构造，而第3.5节讨论构建该有序域所需的 NIF 训练。此外，由于 H 的元素由神经网络生成的向量表示，第3.4节定义了神经序的概念，这对于确立有序域的性质至关重要。

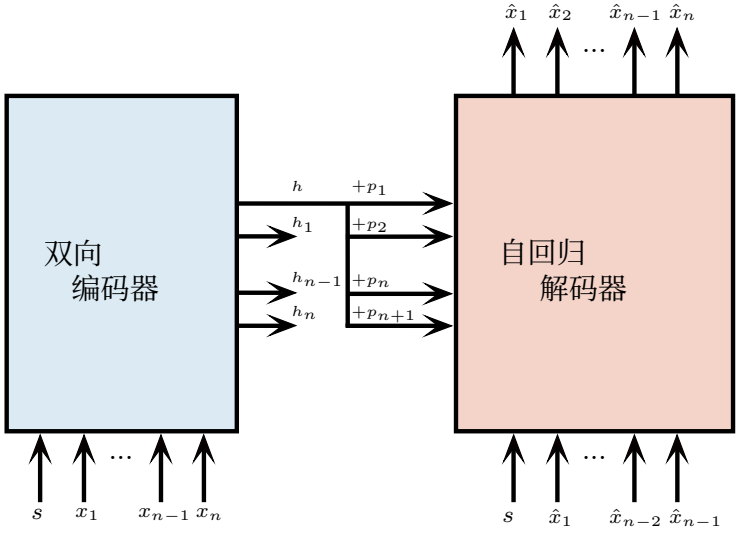


图1：NIF——模型架构。

3.3 阿贝尔神经算子

阿贝尔神经算子（ANO）是一种二元算子：首先，它通过神经网络实现；其次，它具有交换性。该运算符可定义为 $*_\Theta$：$F \times F \to \mathbb{R}^d$。理想情况下，我们希望运算符 $*_\Theta$满足 $\text{Img}(*_\Theta) = \mathbb{Q} *_\Theta \mathbb{Q} = F$，从而确保 $*_\Theta$在 F上是封闭的。

3.3.1 阿贝尔拼接层

阿贝尔拼接（AC）层是一种中间层，旨在使我们的模型能够学习复杂的加法和乘法运算符。在该层中，必须以保持交换性的方式组合输入向量。在 DeepSets 论文中， [48], 证明了在特定条件下，神经网络对集合输入的各种排列保持不变性，这启发了能够保持交换性的层的开发。

为实现这样的层，考虑两个输入向量 h_1， $h_2 \in \mathbb{R}^m$。将这些向量拼接，形成两个新向量： $[h_1, h_2]$ 和 $[h_2, h_1]$。随后，对这些拼接后的向量应用一个非线性函数（相当于一个非线性神经网络），生成两个输出向量 t_1 和 t_2，如公式 (1) 和 (2) 所述。

$$t_1 = f([h_1, h_2]; \theta) \tag{1}$$
$$t_2 = f([h_2, h_1]; \theta) \tag{2}$$

其中， $[.,.]$ 是拼接算子， $f: \mathbb{R}^{2m} \to \mathbb{R}^m$ 是一个非线性函数，相当于一个具有可训练参数 θ的神经网络，例如带有 ReLU 激活函数的全连接层。

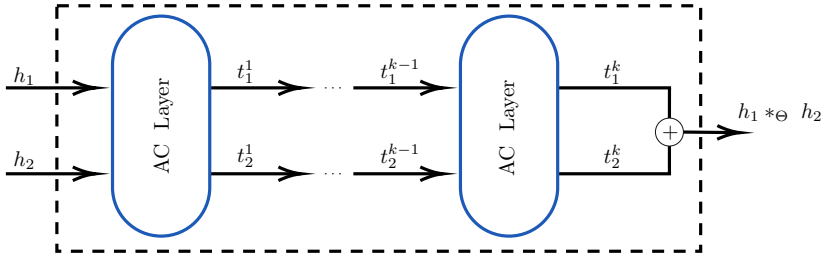


图2：堆叠式阿贝尔神经算子。

3.3.2 堆叠式 AC 层

如图 2 所示，多个 AC 层可以堆叠，以捕捉更复杂的运算符。之所以可以进行堆叠，是因为某一层的输入向量维度与该层的输出向量维度相匹配。堆叠若干层后，最终输出由两个向量组成。这两个向量相加得到最终向量，而这种加法自然保持交换性。

现在考虑一个由 kAC 层构成的序列，其中对于每一层 $i(1 \le i \le k)$，输入为 t_1^{i-1}和 t_2^{i-1}，输出为 t_1^i和 t_2^i，对应的可训练参数记为 θ_i。令 $\Theta = \{\theta_1, \ldots, \theta_k\}$表示所有 k层中的全部可训练参数。对于 h_1， $h_2 \in \mathbb{R}^m$，运算 $h_1 *_\Theta h_2$定义如下:

$$h_1 *_\Theta h_2 = t_1^k + t_2^k = f([t_1^{k-1}, t_2^{k-1}]; \theta_k) + f([t_2^{k-1}, t_1^{k-1}]; \theta_k) \tag{3}$$

其中 $t_1^0 = h_1$， $t_2^0 = h_2$，且对于 $1 \le i \le k$,

$$t_1^i = f(t_1^{i-1}, t_2^{i-1}; \theta_i) \tag{4}$$
$$t_2^i = f(t_2^{i-1}, t_1^{i-1}; \theta_i) \tag{5}$$

3.4 神经序

本节中，我们定义神经排序，它扩展了集合上的序的定义。关键区别在于，在此定义中，排序由神经网络表示。具体而言，神经网络（ω）接收两个嵌入向量作为输入，并产生三种可能状态之一，用以表示这两个向量之间的关系。

假设 $\omega: \mathbb{R}^d \times \mathbb{R}^d \to [0,1]^3$是一个与神经网络等价的函数。对于所有 h_1， $h_2 \in \mathbb{R}^d$，我们按如下方式创建关系 $<_\omega \subseteq \mathbb{R}^d \times \mathbb{R}^d$:

a) $h_1 <_\omega h_2 \iff argmax[\omega(h_1,\ h_2)] = 1$

b) $h_2 <_\omega h_1 \iff argmax[\omega(h_1,\ h_2)] = 2$

c) $h_1 \approx h_2\ (h_1 \not<_\omega h_2,\ h_2 \not<_\omega h_1) \iff argmax[\omega(h_1, h_2)] = 3$

请注意，陈述 (c) 不一定等价于集合上的序定义中的等式条件 (=)。

定义 3.1. 关系 $<_\omega \subseteq \mathbb{R}^d \times \mathbb{R}^d$ 是集合 $\mathbb{R}^d$ 上的神经序，当对所有 h_1、 h_2、 $h_3 \in \mathbb{R}^d$、 $h_1 <_\omega h_2$ 和$h_2 <_\omega h_3$，均有 $h_1 <_\omega h_3$。

命题 3.1。 集合 $\mathbb{R}^d$ 上的一个序是集合 $\mathbb{R}^d$上的一个神经序。

3.5 训练期间保留代数性质

本节探讨如何将 H 转换为与 $\mathbb{Q}$ 同构的有序域。我们考察使 H 成为有序域所需的条件。定理 3.1 展示了实现有序域的理想条件。该定理及其他提及的定理的证明见附录 A。

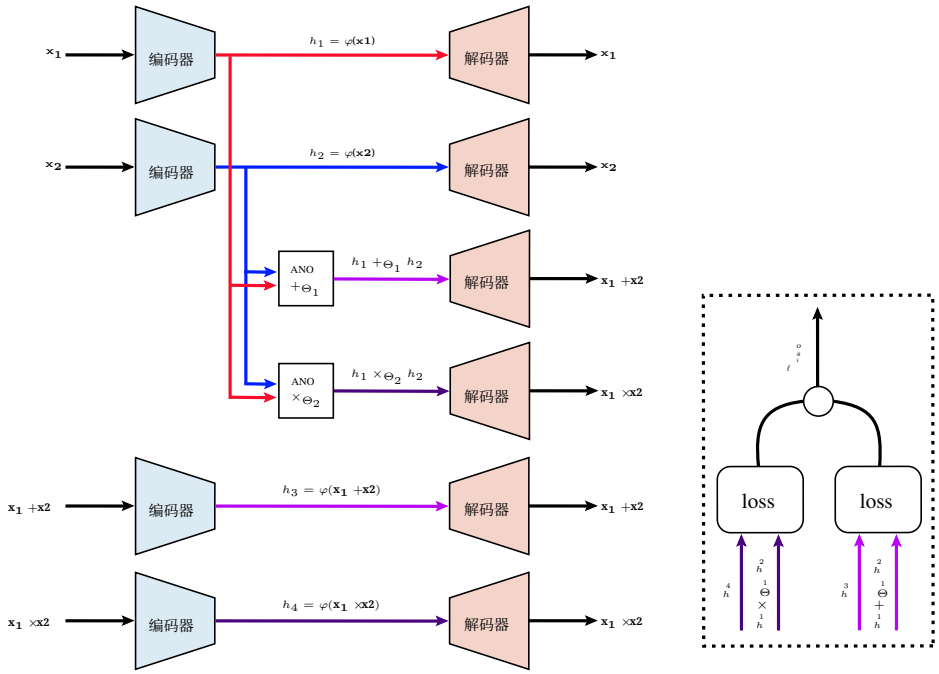


图 3：创建神经同构域的整体训练流程。

定理 3.1. 设 $(F_1, +_1, \times_1, <_1)$ 为一个有序域。对于映射 $\varphi: F_1 \to F_2$、两个运算符 $+_2: \varphi(F_1) \times \varphi(F_1) \to \varphi(F_1)$ 和 $\times_2: \varphi(F_1) \times \varphi(F_1) \to \varphi(F_1)$，以及关系 $<_2$，代数结构 $(\varphi(F_1) +_2, \times_2, <_2)$ 是一个有序域，当且仅当对所有 $a, b \in F_1$，

$$\varphi(a +_1 b) = \varphi(a) +_2 \varphi(b) \tag{6}$$
$$\varphi(a \times_1 b) = \varphi(a) \times_2 \varphi(b) \tag{7}$$
$$a <_1 b \Leftrightarrow \varphi(a) <_2 \varphi(b) \tag{8}$$

定理 3.1 中的条件 6、7 和 8 确保了有序域的建立。为了在 NIF 模型中满足这些条件，可以定义两个损失函数：损失函数 ℓ_{iso} 针对条件 6 和 7，而损失函数 ℓ_{ord}针对条件 8。为了确定使 H 与 $\mathbb{Q}$ 同构的理想条件，定理 3.2、3.3 和 3.4 探讨了必要准则。这些定理还给出了 ℓ_{iso} 和 ℓ_{ord} 的定义。此外，实现同构还需要损失函数 ℓ_{rec}，它表示自编码器神经网络中的重构损失，其定义也在文中给出。图 3 展示了创建神经同构域的整体训练流程。

定理 3.2。 函数 φ是一一映射，当对于一个度量 d，

$$\ell_{rec} = \mathbb{E}_{a \sim D}[d(\varphi^{\dagger}(\varphi(a)), a)] = 0 \tag{9}$$

定理 3.3。 $<_\omega$ 是定义在 $\varphi(\mathbb{Q})$上的神经序，当

$$\ell_{ord} = \mathbb{E}_{a,b \sim D}[[1(a < b) - 1(\varphi(a) <_\omega \varphi(b))]^2] = 0 \tag{10}$$

定理 3.4。 代数结构 $(\varphi(\mathbb{Q}) \dot{+}, \dot{\times}, <_\omega)$ 是有序域，且与 $(\mathbb{Q}, +, \times, <)$ 同构，当

$$\ell_{total} = \ell_{rec} + \ell_{iso} + \ell_{ord} = 0 \tag{11}$$

其中

$$\ell_{rec} = \mathbb{E}_{a\sim D}[d(\varphi^{\dagger}(\varphi(a)), a)] \tag{12}$$
$$\ell_{ord} = \mathbb{E}_{a,b\sim D}[(1(a < b) - 1(\varphi(a) <_{\omega} \varphi(b)))^2] \tag{13}$$
$$\ell_{iso} = \mathbb{E}_{a,b\sim D}[d(\varphi(a+b),\ \varphi(a) \dot{+} \varphi(b))] \tag{14}$$
$$+ \mathbb{E}_{a,b\sim D}[d(\varphi(a\times b),\ \varphi(a) \dot{\times} \varphi(b))] \tag{15}$$

条件 56 足以构造一个与有理数域同构的域；然而，这在实践中无法实现。为了更好地满足这一条件，我们需要选择合适的模型和训练方法。

4 评估指标

据我们所知，这是首个提出构造能够保留代数性质的数字嵌入方法的研究。因此，有必要建立具体的评估标准，以评估该模型的性能。本节介绍用于衡量模型在数字重构、加法和乘法方面准确率的内部评估指标。

- **单位元测试：**单位元测试标准用于确定模型在其中一个操作数为单位元时执行加法和乘法运算的能力。
- **可逆性测试：**此测试用于确定一个数的嵌入与该数逆元的嵌入进行加法和乘法运算后，是否会得到单位元。
- **封闭性测试：**此测试用于确定模型所构建的空间具有多大程度的封闭性。为此，我们可以检验模型能否重构向量空间中两个嵌入数字的加法和乘法结果。单位元测试和可逆性测试是此测试的两种特殊模式。
- **结合性测试：** 该标准规定了模型是否能够保持结合律。为了进行研究，只需按照两种不同的括号顺序，对三个数字的嵌入进行加法和乘法运算即可。
- **分配律测试：** 该标准与结合性测试标准相同，不同之处在于，它考察的不是结合律，而是分配律。

5 实验

5.1 分布与数值数据集

为了使用数值数据训练模型，我们定义了一个整数分布，并用其进行采样。要生成一个有理数，需要从该分布中抽取两个样本：第一个样本决定整数部分的长度，第二个样本决定小数部分的长度。随后，根据各自的长度，使用均匀分布随机生成整数部分和小数部分。

从理论上讲，关于数字长度的概率分布应当能够生成任意有理数。由于数字长度是可以无限增加的离散变量，因此所使用的分布也应当是离散的，并覆盖区间 $[0, +\infty)$ [2]。然而，受计算能力限制，在实践中，生成极大或极小数字（无论正负）的概率应当较低。此外，我们的实证结果表明，如果生成的长度不偏向较小的取值，测试性能会显著下降。这主要是因为两个极大或极小的数字相乘后，所得数字会显著大于或小于数据中出现的数字，从而可能对乘法运算的结果产生非常不利的影响。偏向较短长度的另一个原因是，加法单位元和乘法单位元都表示为个位数，这会导致单位元测试及其他测试的结果普遍下降。因此，适当的分布应当是离散的，覆盖范围 $[0, +\infty)$，并且偏向较小的数字。

数字长度的候选分布包括：泊松分布、负二项分布和几何分布（对于几何分布，从每个采样长度中减去一个单位，以计入长度为

[2]请注意，若要省略整数部分或小数部分，长度必须为零；如果两部分的长度都为零，则会重新采样。

零的情况）。通过调整这三种分布的参数，我们考察了 30 多种不同的分布，其中部分分布如图4所示。在实证实验中，我们分析了短、中、长数字长度的偏差，结果表明，偏向较短数字长度的分布在代数指标上取得了显著更好的结果。对于每种分布，我们完全独立地生成了 3000 万个训练样本和 100 万个评估样本。负二项分布在参数 $r = 2$ 和 $p = 0.45$ 时取得了最佳结果。

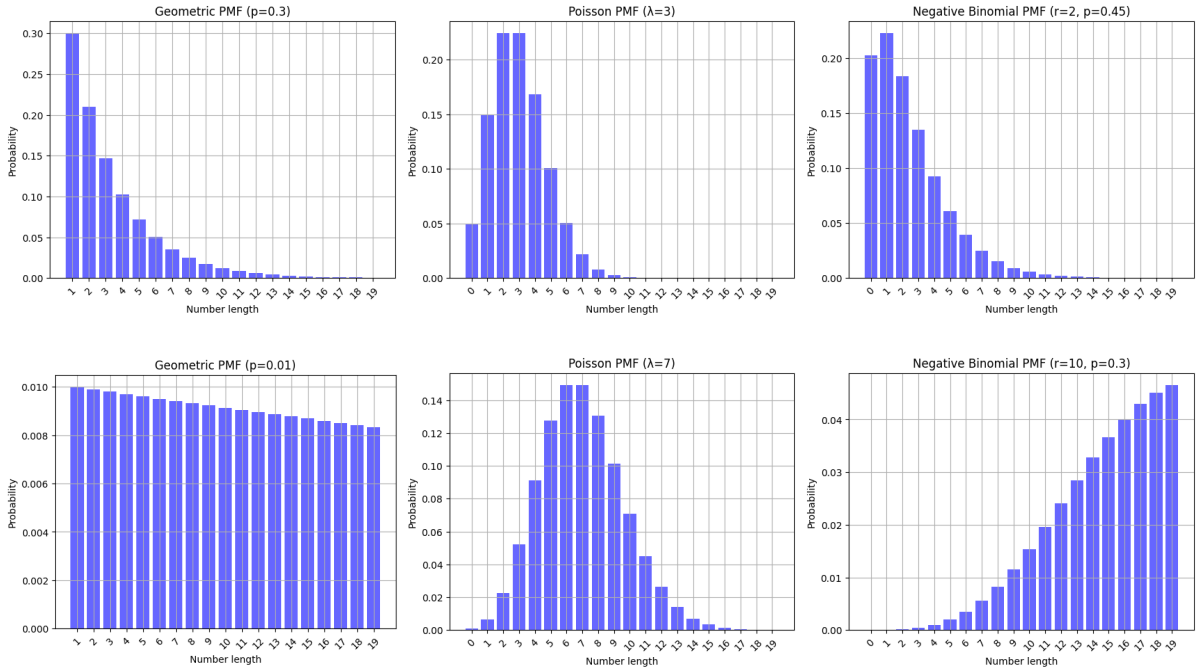


图4：所测试的数字长度分布的一部分

5.2 实验设置

序列中的每个位数都表示为一个维度为14的独热编码向量，其中包括0到9的位数、小数点以及正负号。对隐藏维度从64到4096不等的Transformer层进行了评估，并在0到5的范围内测试了每个运算符的层数。层数为零表示直接对嵌入向量采用简单的向量加法或Hadamard乘积。

还测试了128、256和512的批次大小，其中批次大小为512时取得了最佳结果。所使用的优化器为Adam，学习率调度如图5所示。Transformer模型由4层组成，每层包含8个注意力头。

训练在配备12GB显存的笔记本GPU RTX4080上进行。每个训练轮次大约需要15分钟，模型共训练了20个训练轮次。

5.3 结果

5.3.1 训练与验证

鉴于神经场包含加法和乘法这两个运算符，图6b报告了模型在每个周期中对训练数据和评估数据执行这两个运算符时的重建准确率。然而，为了分别评估每个运算符，我们创建了加法群和乘法组，并分别评估仅包含加法运算符的模型和仅包含乘法运算符的模型的重建准确率。该评估的结果如图6a所示。

结果表明，首先，模型在训练数据和评估数据上的表现非常相似。其次，与加法组相比，域在加法运算符上的表现较差；但与乘法组相比，域在乘法运算符上的表现较好。

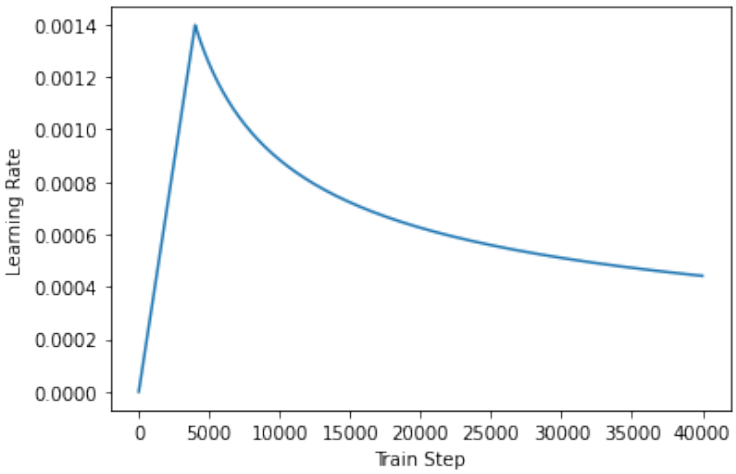


图5：Adam 优化器的学习率调度

5.3.2 消融研究

在训练和评估模型之前，我们必须确定使模型性能达到最佳的最大数字长度、理想的嵌入向量长度（隐藏维度），以及每个运算符的层数。这些超参数的结果见图 7、8a 和 8b。

我们从结果中得出结论：如果数字的长度在20至25位之间，我们可以向模型提供相对较大的数字，并获得令人满意的性能。此外，如果模型的维度超过512，模型性能不会有显著提升。因此，我们将模型的维度设为512。

对于每个运算符，我们还考察了层数从0到6的情况，发现层数超过1层会导致重建准确率下降。这是假设每次实验的训练轮数保持不变的情况。然而，如果增加层数的同时也增加训练轮数，重建准确率就会提高。尽管如此，增加训练轮数会带来沉重的计算负荷，因此我们只采用一层。

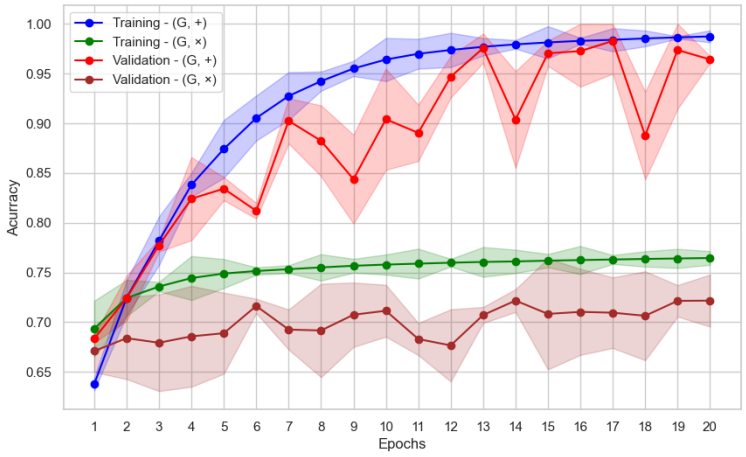
5.3.3 代数指标

第 4 节定义的不同实验结果见表2和表3。表2分别展示加法和乘法运算符的结果，而表3展示联合测试的结果。

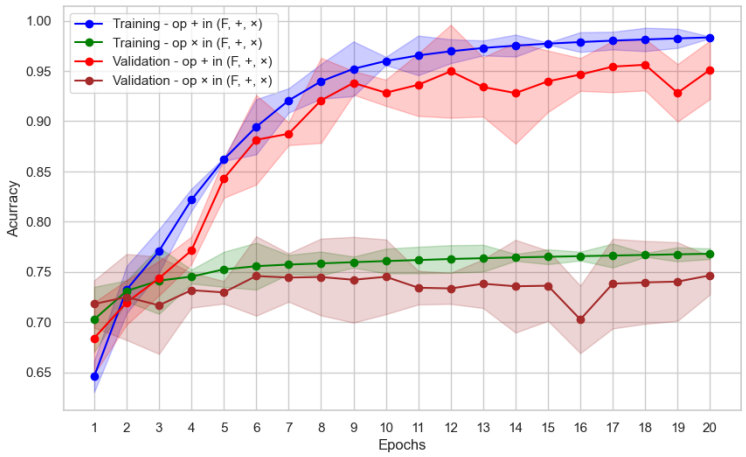
表2：加法运算的单位元测试、封闭性测试、可逆性测试和结合性测试的结果 n 和乘法

		准确率	精确匹配	总损失
加法	**单位元测试**	98.95 ± 0.08	86.65 ± 0.12	0.002 ± 0.0003
	封闭性测试	99.21 ± 0.13	89.70 ± 0.11	0.002 ± 0.0005
	可逆性测试	97.77 ± 0.31	82.59 ± 0.44	0.004 ± 0.0009
	结合性测试	95.86 ± 0.35	79.32 ± 0.69	0.004 ± 0.0008
乘法	**单位元测试**	61.25 ± 1.29	38.12 ± 4.33	0.007 ± 0.0012
	封闭性测试	73.22 ± 0.89	40.31 ± 2.92	0.007 ± 0.0010
	可逆性测试	56.66 ± 1.53	37.78 ± 3.22	0.009 ± 0.0022
	结合性测试	53.55 ± 0.96	34.19 ± 3.51	0.009 ± 0.0030

加法和乘法的代数测试结果显示，两者的表现截然不同。加法结果表明，其满足代数性质的效果非常出色。单位元、封闭性、可逆性和结合性测试的准确率均超过 95%，其中单位元测试达到 98.95%，封闭性测试达到 99.21%。这些结果凸显了加法满足这些基本性质的稳健性和可靠性，这一点也体现在极低的总损失值上。



(a)



(b)

Fig图 6：神经群和神经场模型在训练集和评估数据集上的准确率 s

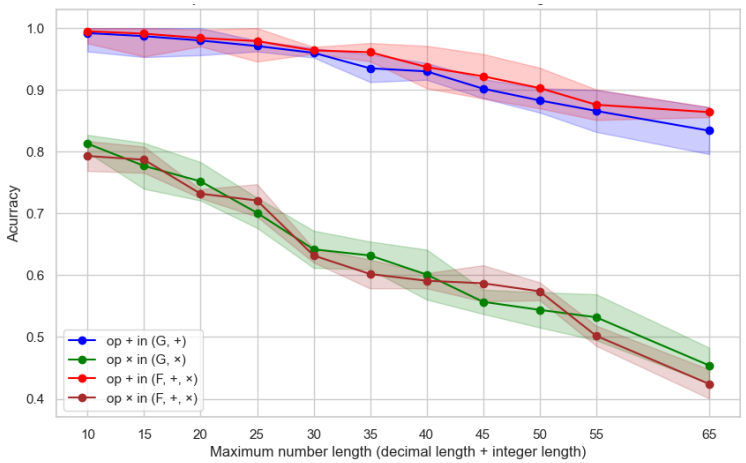
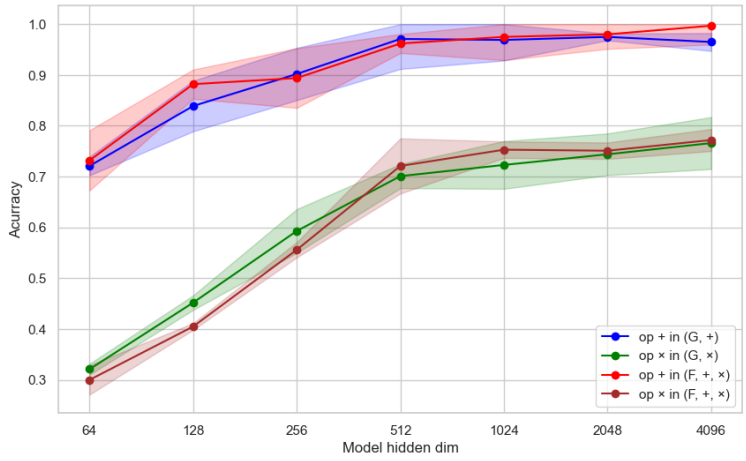
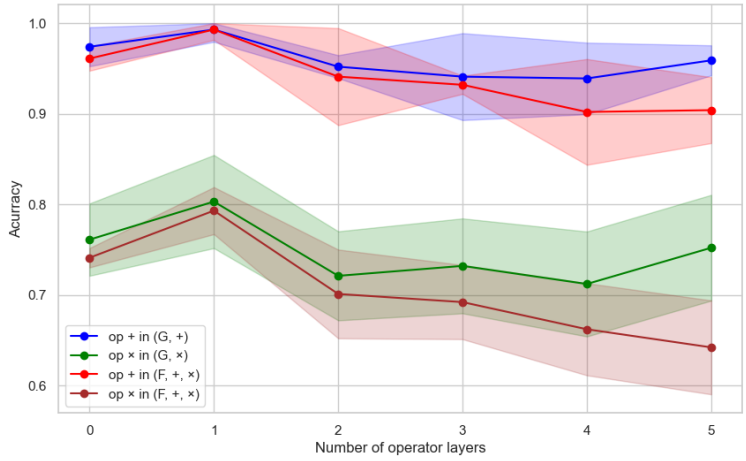


图 7：数字长度对模型准确率的影响



(a)



(b)

图8：隐藏神经元的数和层数对模型准确率的影响

表3：分配律测试和顺序测试的结果

	准确率	精确匹配	总损失
分配律测试	52.97 ± 3.74	29.99 ± 4.12	0.011 ± 0.0056
顺序测试	97.25 ± 0.024	–	–

相比之下，乘法的结果表现明显较差。分配律测试和顺序测试的准确率显著较低，介于 53.55% 至 73.22% 之间，其中分配律测试尤其困难，准确率仅为 52.97%。这表明模型在满足分配律方面存在较大困难。乘法较高的总损失值进一步反映出错误更为普遍。虽然顺序测试的准确率相对较高，达到 97.25%，但缺乏更多细节，限制了全面评估。总体而言，与加法相比，乘法在满足代数性质方面表现较差，凸显了有待改进的领域。

总而言之，加法准确率高、损失小，在代数运算中表现可靠。相比之下，乘法难以达到类似的性能水平。加法和乘法算子性能存在差异的原因之一在于操作数的位数。如果一个操作数有 n 位，另一个有 m 位（其中 $m \geq n$），那么这两个操作数的和将有 m 位或 $m+1$ 位。

相比之下，它们的乘积将有 $m+n-1$ 位或 $m+n$ 位。因此，由于训练期间自编码器未接触从 m 位数到 2m 位数这一范围内的数字，乘法运算可以视为一个外推问题，而加法则可以视为一个插值问题。

5.3.4 与现有工作比较

据我们所知，现有文献中尚无工作提出一种构建数字嵌入的方法，明确旨在保留加法、乘法及其相关特性（例如单位元、可逆性、封闭性、结合律和分配律）等代数性质。因此，无法使用代数指标将所提出的嵌入与先前方法直接进行比较。尽管如此，为了比较我们的工作，我们采用第2.1节中提到的嵌入方法 LUNA [42], FoNE [2], 和 xVal [3] 作为基线方法，比较结果如表4所示。我们通过在不同长度下，使用这些嵌入方法评估加法和乘法运算的准确率与精确匹配率来进行比较。

为确保比较公平且一致，采用第5.1节所述的数据分布来生成用于训练嵌入向量的数据集。此外，由于所提出的数字嵌入方法不需要编码器部分，只需要解码器、加法和乘法算子（加法和乘法算子，ANOs）以及神经序模块，因此针对每种嵌入仅训练这些组件。如表4所示，NIF在所有任务和输入长度下均持续优于基线方法。此外，NIF取得了最高的准确率和精确匹配率，略高于 LUNA 和 FoNE 等强大模型。在乘法任务中，尽管 LUNA 的表现相对良好，NIF 仍保持更高的准确率和精确匹配率，尤其是在数字长度增加时。对于序关系任务，NIF 和 LUNA 均表现出色，其中 NIF 在所有设置下均略优于 LUNA。值得注意的是，xVal 的性能出现了显著下降，尤其是在处理较长数字时。

表4：性能　　不同模型在不同最大数字长度下的加法、乘法和序关系任务　　r
长度（10、20 和 3　0）的结果。结果以准确率和精确匹配（%）表示。

	模型	最大长度=10		最大长度=20		最大长度=30	
		准确率	精确匹配	准确率	精确匹配	准确率	精确匹配
加法	LUNA [42]	99.47±0.31	90.45±0.48	99.30±0.17	91.96±0.42	97.04±0.36	79.28±0.79
	FoNE [2]	98.97±0.51	86.94±0.82	98.58±0.68	82.97±0.51	97.84±0.36	73.78±0.99
	xVal [3]	32.15±1.14	13.87±3.14	9±2.14	1.14±0.99	0	0
	NIF（我们的方法）	99.91±0.11	93.14±0.21	99.25±0.22	90.71±0.33	98.99±0.25	87.12±0.44
乘法	LUNA	84.63±2.16	61.45±1.51	75.42±2.14	40.87±2.87	69.97±1.77	22.87±1.56
	FoNE	71.76±1.06	32.88±2.16	65.49±1.44	20.84±2.74	59.37±2.95	12.24±2.67
	xVal	8.44±3.12	0.14±0.09	0	0	0	0
	NIF（我们的方法）	82.15±0.79	59.18±1.33	78.64±0.85	48.54±1.99	75.09±0.89	41.18±2.22
序	LUNA	98.13±0.25	–	97.24±0.49	–	96.17±0.55	–
	FoNE	96.06±0.44	–	94.47±0.26	–	93.78±0.88	–
	xVal	28.18±1.25	–	17.55±2.18	–	7.65±2.66	–
	NIF（我们的方法）	98.14±0.16	–	97.80±0.21	–	97.43±0.33	–

6 结论

在本研究中，我们提出了一种构建数字嵌入的新方法，旨在保留代数性质，重点关注加法和乘法运算。我们还提出了专门的评估指标，用于评估模型维持这些性质的能力。实验结果表明，模型在不同算术运算上的表现存在明显差异。加法运算表现出色，所有测试的准确率均超过 95%。这表明模型能够有效保留与加法相关的代数性质。相较之下，乘法运算的表现明显较差，准确率介于 53.55% 至 73.22% 之间。这种差异凸显了在乘法运算中实现同等程度代数性质保留的挑战。

总体而言，我们在嵌入数字并保留其代数性质方面的方法展现出良好前景，尤其是在加法运算中。然而，乘法的结果表明，仍需进一步改进，以提升模型在更复杂运算中保持代数一致性的能力。

7 未来工作

未来的工作将着重解决当前方法中观察到的局限性，尤其是与乘法相关的挑战。可供改进和探索的几个方向包括：

- **增强乘法嵌入：**研究专门用于捕捉乘法复杂性的更精细嵌入技术，例如使用更深层的网络或其他神经网络架构。
- **替代分布：**探索不同的数据分布，以生成更能代表模型将遇到的数字和运算范围的训练数据集。这或许有助于提升模型的泛化能力和性能，尤其是在乘法方面。
- **扩展代数运算：**扩大评估范围，纳入除法和乘方等其他代数运算，以评估模型在更广泛的运算范围内保留代数性质的多功能性和鲁棒性。
- **模型优化：**采用超参数调优和自适应学习率等先进优化技术，以提升模型的收敛性和准确率，尤其是在保留更复杂运算的性质方面。
- **现实应用：**将开发出的嵌入应用于科学计算、密码学或金融建模等实际任务，在这些任务中，保留代数性质至关重要。这有助于深入了解所提出方法的实用价值及其局限性。

通过探索这些方向，我们旨在进一步提升数字嵌入保留代数性质的能力，并扩大其在各个领域的适用范围。

参考文献

[1] Chengyue Jiang、Zhonglin Nian、Kaihao Guo、Shanbo Chu、Yinggong Zhao、Libin Shen 和 Kewei Tu。学习数字嵌入。发表于*EMNLP*，2020年。[2] Tianyi Zhou、Deqing Fu、Mahdi Soltanolkotabi、Robin Jia 和 Vatsal Sharan。FoNE：通过傅里叶特征实现精确的单词元数字嵌入，2025年。[3] Siavash Golkar、Mariel Pettee、Michael Eickenberg，……和 Shirley Ho。xVal：面向大语言模型的连续数字编码。发表于*NeurIPS*，2023年。[4] Dhanasekar Sundararaman、Shijing Si、Vivek Subramanian、Guoyin Wang、Devamanyu Hazarika 和 Lawrence Carin。保留数值能力的词嵌入方法。发表于*EMNLP*，2020年。[5] Tomas Mikolov、Kai Chen、Gregory S. Corrado 和 Jeffrey Dean。向量空间中词表示的高效估计。发表于*ICLR*，2013年。[6] Jeffrey Pennington、Richard Socher 和 Christopher Manning。GloVe：词表示的全局向量。发表于*EMNLP*，2014年。[7] Piotr Bojanowski、Edouard Grave、Armand Joulin 和 Tomas Mikolov。利用子词信息丰富词向量。*TACL*，2017年。[8] Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。BERT：用于语言理解的深度双向 Transformer 预训练。发表于*NAACL-HLT*，第4171–4186页，2019年。[9] Zhilin Yang、Zihang Dai、Yiming Yang、Jaime Carbonell、Russ R Salakhutdinov 和 Quoc V Le。XLNet：用于语言理解的通用自回归预训练。发表于*NeurIPS*，第32卷，2019年。[10] Mike Lewis、Yinhan Liu、Naman Goyal、Marjan Ghazvininejad、Abdelrahman Mohamed、Omer Levy、Veselin Stoyanov 和 Luke Zettlemoyer。BART：用于自然语言生成、翻译和理解的去噪序列到序列预训练。发表于*ACL*，2020年。[11] Kevin Clark、Minh-Thang Luong、Quoc V. Le 和 Christopher D. Manning。ELECTRA：将文本编码器预训练为判别器，而非生成器。发表于*ICLR*，2020年。[12] Zhenzhong Lan、Mingda Chen、Sebastian Goodman、Kevin Gimpel、Piyush Sharma 和 Radu Soricut。ALBERT：用于语言表示自监督学习的轻量级 BERT。发表于*ICLR*，2020年。[13] Zhengyan Zhang、Xu Han、Zhiyuan Liu、Xin Jiang、Maosong Sun 和 Qun Liu。ERNIE：融合信息性实体的增强语言表示。发表于*ACL*，2019年。

[14] Colin Raffel、Noam Shazeer、Adam Roberts、Katherine Lee、Sharan Narang、Michael Matena、Yanqi Zhou、Wei Li 和 Peter J. Liu。借助统一的文本到文本 Transformer 探索迁移学习的极限。机器学习研究期刊，21(140)：1–67，2020年。[15] Manzil Zaheer、Guru Guruganesh、Kumar Avinava Dubey、Joshua Ainslie、Chris Alberti、Santiago Ontanon、Philip Pham、Anirudh Ravula、Qifan Wang、Li Yang 和 Amr Ahmed。Big Bird：面向更长序列的 Transformer。发表于 NeurIPS，第33卷，2020年。[16] Pengcheng He、Xiaodong Liu、Jianfeng Gao 和 Weizhu Chen。DeBERTa：采用解耦注意力的解码增强型 BERT。发表于 ICLR，2021年。[17] Will Hamilton、Zhitao Ying 和 Jure Leskovec。大型图上的归纳式表征学习。发表于 NeurIPS，第30卷，2017年。[18] Thomas N. Kipf 和 Max Welling。利用图卷积网络进行半监督分类。发表于 ICLR，2017年。[19] Petar Veli ˇckovi ´c、Guillem Cucurull、Arantxa Casanova、Adriana Romero、Pietro Liò 和 Yoshua Bengio。图注意力网络。发表于 ICLR，2018年。[20] Keyulu Xu、Weihua Hu、Jure Leskovec 和 Stefanie Jegelka。图神经网络的能力有多强？发表于 ICLR，2019年。[21] Yuning You、Tianlong Chen、Yongduo Sui、Ting Chen、Zhangyang Wang 和 Yang Shen。通过增强进行图对比学习。发表于 NeurIPS，第33卷，第5812–5823页，2020年。[22] Andrew Trask、Felix Hill、Scott E Reed、Jack Rae、Chris Dyer 和 Phil Blunsom。神经算术逻辑单元。发表于 NeurIPS，第31卷，2018年。[23] Andreas Madsen 和 Alexander Rosenberg Johansen。神经算术单元。发表于 ICLR，卷号 abs/2001.05016，2020年。[24] Lukasz Kaiser 和 Ilya Sutskever。神经图形处理器学会算法。发表于 ICLR，2016年。[25] Wojciech Zaremba 和 Ilya Sutskever。学习执行，2014年。arXiv:1410.4615。[26] Georgios P. Spithourakis 和 Sebastian Riedel。语言模型的数值能力：评估并提升其预测数字的能力。发表于 ACL，2018年。[27] I.N. Herstein。《抽象代数》。普伦蒂斯霍尔出版社，1996年。[28] W. Rudin。《数学分析原理》。麦格劳-希尔，1976年。[29] S.Y.T. Lin 和 Y.F. Lin。《集合论及其应用》。图书出版商，1985年。[30] Xikun Zhang、Deepak Ramachandran、Ian Tenney、Yanai Elazar 和 Dan Roth。语言嵌入能否捕捉尺度？发表于 EMNLP，2020年。[31] Philip Gage。一种新的数据压缩算法。《C Users J.》，12(2)：23–38，1994年。[32] Yonghui Wu、Mike Schuster、Zhifeng Chen，……和 Jeffrey Dean。谷歌神经机器翻译系统：弥合人工翻译与机器翻译之间的差距，2016年。arXiv:1609.08144。[33] Taku Kudo 和 John Richardson。SentencePiece：一种简单且独立于语言的子词分词器和反分词器，用于神经文本处理。发表于 EMNLP，2018年。[34] Alec Radford、Karthik Narasimhan、Tim Salimans 和 Ilya Sutskever。通过无监督学习提升语言理解能力。技术报告，OpenAI，2018年。[35] Alec Radford、Jeff Wu、Rewon Child、David Luan、Dario Amodei 和 Ilya Sutskever。语言模型是无监督的多任务学习器。技术报告，OpenAI，2019年。[36] Tom Brown、Benjamin Mann、Nick Ryder，……和 Dario Amodei。语言模型是少样本学习器。发表于 NeurIPS，第33卷，第1877–1901页，2020年。[37] Hugo Touvron、Thibaut Lavril、Gautier Izacard，……和 Guillaume Lample。Llama：开放且高效的基础语言模型，2023年。[38] Hugo Touvron、Louis Martin、Kevin Stone，……和 Thomas Scialom。Llama 2：开放的基础语言模型和微调聊天模型，2023年。[39] Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Ł ukasz Kaiser 和 Illia Polosukhin。注意力机制就是你所需要的一切。发表于 NeurIPS，第5998–6008页，2017年。[40] Rewon Child、Scott Gray、Alec Radford 和 Ilya Sutskever。利用稀疏 Transformer 生成长序列，2019年。

[41] Iz Beltagy、Matthew E. Peters 和 Arman Cohan。Longformer：长文档 Transformer，2020年。[42] Hongwei Han、Jialiang Xu、Mengyu Zhou、Yijia Shao、Shi Han 和 Dongmei Zhang。LUNA：通过数字插件和预训练，在 Transformer 上利用数字增强实现语言理解，2023年。[43] Jasivan Alex Sivakumar 和 Nafise Sadat Moosavi。如何利用数字嵌入来表示数字？载于*COLING*，2025年。[44] Leonardo Franco 和 Sergio A. Cannas。使用前馈神经网络求解算术问题。*Neurocomputing*，18(1)：61–79，1998年。[45] Sean McLeish、Arpit Bansal、Alex Stein、.....和 Tom Goldstein。只要嵌入得当，Transformer 就能进行算术运算。载于*NeurIPS*，2024年。[46] Alejandro Parada-Mayorga 和 Alejandro Ribeiro。代数神经网络：形变稳定性。*IEEE*信号处理汇刊，69：3351–3366，2021年。[47] R. Zimmer。神经网络代数理论：序贯复合，第399–403页。施普林格，1997年。[48] Manzil Zaheer、Satwik Kottur、Siamak Ravanbakhsh、Barnabas Poczos、Russ R Salakhutdinov 和 Alexander J Smola。深度集合。载于*NeurIPS*，第30卷，2017年。

定理证明

引理 A.1. 设$(G_1, *_1, e)$ 为一个群（阿贝尔群）。对于映射 $\varphi: G_1 \to G_2$ 和运算 $*_2: \varphi(G_1) \times \varphi(G_1) \to \varphi(G_1)$，代数结构$(\varphi(G_1) *_2, \varphi(e))$ 是一个群（阿贝尔群），当且仅当对于所有$a, b \in G_1$

$$\varphi(a *_1 b) = \varphi(a) *_2 \varphi(b). \tag{16}$$

证明。 1. $*_2$ 在 $\varphi(G_1)$上封闭，因为对于所有 $a, b \in G_1$,

$$\varphi(a) *_2 \varphi(b) = \varphi(a *_1 b) \in \varphi(G_1). \tag{17}$$

2. $*_2$ 在 $\varphi(G_1)$上满足结合性，因为对于所有 $a, b, c \in G_1$,

$$\begin{aligned}
\varphi(a) *_2 (\varphi(b) *_2 \varphi(c)) &= \varphi(a) *_2 \varphi(b *_1 c) && (18)\\
&= \varphi(a *_1 (b *_1 c)) && (19)\\
&= \varphi((a *_1 b) *_1 c) && (20)\\
&= \varphi(a *_1 b) *_2 \varphi(c) && (21)\\
&= (\varphi(a) *_2 \varphi(b)) *_2 \varphi(c). && (22)
\end{aligned}$$

3. $\varphi(e) \in \varphi(G_1)$是单位元，因为对于所有 $a \in G_1$,

$$\begin{aligned}
\varphi(a) *_2 \varphi(e) &= \varphi(a *_1 e) = \varphi(a), && (23)\\
\varphi(e) *_2 \varphi(a) &= \varphi(e *_1 a) = \varphi(a). && (24)
\end{aligned}$$

4. 对于所有 $a \in G_1$， $\varphi(a^{-1})$ 是 $\varphi(a)$ 的逆元，因为

$$\begin{aligned}
\varphi(a) *_2 \varphi(a^{-1}) &= \varphi(a *_1 a^{-1}) = \varphi(e), && (25)\\
\varphi(a^{-1}) *_2 \varphi(a) &= \varphi(a^{-1} *_1 a) = \varphi(e). && (26)
\end{aligned}$$

5. 如果 $(G_1, *_1)$ 是阿贝尔的，那么 $(\varphi(G_1) *_2)$ 也是阿贝尔的，因为对于所有 $a, b \in G_1$,

$$\begin{aligned}
\varphi(a) *_2 \varphi(b) &= \varphi(a *_1 b) && (27)\\
&= \varphi(b *_1 a) && (28)\\
&= \varphi(b) *_2 \varphi(a). && (29)
\end{aligned}$$

■

定理 A.2. *设 $(F_1, +_1, \times_1)$ 是一个域。对于映射 $\varphi: F_1 \to F_2$、两个运算符 $+_2: \varphi(F_1) \times \varphi(F_1) \to A_1$ 和 $\times_2: \varphi(F_1) \times \varphi(F_1) \to A_2$，代数结构 $(\varphi(F_1) +_2, \times_2)$ 是一个域，如果对于所有 $a, b \in F_1$,*

$$\varphi(a +_1 b) = \varphi(a) +_2 \varphi(b), \tag{30}$$
$$\varphi(a \times_1 b) = \varphi(a) \times_2 \varphi(b). \tag{31}$$

证明。由引理 A.1，我们有 $A_1 = A_2 = \varphi(F_1)$，且 $(\varphi(F_1) +_2, \varphi(0))$ 和 $(\varphi(F_1) \setminus \{\varphi(0)\}, \times_2, \varphi(1))$ 是阿贝尔群。此外，对于所有 a，b，$c \in F_1$，我们有

$$\begin{aligned}
\varphi(a) \times_2 (\varphi(b) +_2 \varphi(c)) &= \varphi(a) \times_2 \varphi(b +_1 c) \\
&= \varphi(a \times_1 (b +_1 c)) \\
&= \varphi(a \times_1 b +_1 a \times_1 c) \\
&= \varphi(a \times_1 b) +_2 \varphi(a \times_1 c) \\
&= \varphi(a) \times_2 \varphi(b) +_2 \varphi(a) \times_2 \varphi(c).
\end{aligned}$$

■

引理 A.3。 *设 $<_1$ 是集合 A_1 上的一个序。对于映射 φ： $A_1 \to A_2$，关系 $<_2 \subseteq \varphi(A_1) \times \varphi(A_1)$ 是一个序，当且仅当对于所有 $a, b \in A_1$*

$$a <_1 b \iff \varphi(a) <_2 \varphi(b) \tag{32}$$

证明。假设存在 a，$b \in A_1$，使得 $\varphi(a) = \varphi(b)$ 且 $a \neq b$。这意味着 $a <_1 b$ 或 $b <_1 a$。因此，$\varphi(a) <_2 \varphi(b)$ 或 $\varphi(b) <_2 \varphi(a)$。不失一般性，假设 $\varphi(a) <_2 \varphi(b)$。

$$\varphi(a) <_2 \varphi(b) = \varphi(a) \implies \tag{33}$$
$$\varphi(a) <_2 \varphi(a) \iff \tag{34}$$
$$a <_1 a \tag{35}$$

这是矛盾。因此,

$$a = b \iff \varphi(a) = \varphi(b) \tag{36}$$

因此，以下陈述中有且仅有一个

$$a <_1 b \iff \varphi(a) <_2 \varphi(b), \tag{37}$$
$$b <_1 a \iff \varphi(b) <_2 \varphi(a), \tag{38}$$
$$a = b \iff \varphi(a) = \varphi(b) \tag{39}$$

为真。

此外，若 $a, b, c \in A_1$ 且 $\varphi(a) <_2 \varphi(b)$ 和 $\varphi(b) <_2 \varphi(c)$，则 $a <_1 b$ 且 $b <_1 c$。因此， $a <_1 c$ 这意味着，$\varphi(a) <_2 \varphi(c)$。
■

推论 A.4. *设 $<_1$ 是集合 A_1 上的一个序，且 $\varphi: A_1 \to A_2$ 是一个映射。对于关系 $<_2 \subseteq \varphi(A_1) \times \varphi(A_1)$，$\varphi$ 是一一的，当且仅当对所有 $a, b \in A_1$*

$$a <_1 b \iff \varphi(a) <_2 \varphi(b)$$

定理 A.5. *设 $(F_1, +_1, \times_1, 0, 1, <_1)$ 是一个有序域。对于映射 $\varphi: F_1 \to F_2$，两个运算符$+_2: \varphi(F_1) \times \varphi(F_1) \to \varphi(F_1)$ 和 $\times_2: \varphi(F_1) \times \varphi(F_1) \to \varphi(F_1)$，以及关系 $<_2$，代数结构 $(\varphi(F_1) +_2, \times_2, \varphi(0)\ \varphi(1)\ <_2)$ 是一个有序域，当且仅当对所有 a, $b \in F_1$*

$$\varphi(a +_1 b) = \varphi(a) +_2 \varphi(b)$$
$$\varphi(a \times_1 b) = \varphi(a) \times_2 \varphi(b)$$
$$a <_1 b \Leftrightarrow \varphi(a) <_2 \varphi(b)$$

证明。 根据定理 A.2，由于 $(F_1, +_1, \times_1, 0, 1)$ 是一个域，因此 $(\varphi(F_1) +_2, \times_2, \varphi(0)\ \varphi(1))$ 也是一个域。根据引理 A.3，由于 $<_1$ 是集合 F_1上的一个序，因此 $<_2$ 是集合 $\varphi(F_1)$上的一个序。现在我们证明 $\varphi(F_1)$ 是一个有序域。

a) 对于 $a, b, c \in F_1, \varphi(b) <_2 \varphi(c)$ 蕴含 $b <_1 c$，然后 $a +_1 b <_1 a +_1 c$。因此,

$$\varphi(a) +_2 \varphi(b) = \varphi(a +_1 b) <_2 \varphi(a +_1 c) = \varphi(a) +_2 \varphi(c) \tag{40}$$

b) 假设 $\varphi(0) <_2 \varphi(a)$ 且 $\varphi(0) <_2 \varphi(b)$，其中 $a, b \in F_1$。 $\varphi(0) <_2 \varphi(a)$ 蕴含 $0 <_1 a$，且 $\varphi(0) <_2 \varphi(b)$ 蕴含$0 <_1 b$。因此 $0 <_1 a \times_1 b$。所以 $0 <_2 \varphi(a) \times_2 \varphi(b)$。

■

定理 A.6. 设 $(\mathbb{Q}, *)$ 是一个群（阿贝尔群）。$(\varphi_{\Theta_E}(\mathbb{Q}) *_\Theta)$ 是一个群（阿贝尔群），当 $meterd : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$ 时，

$$\ell_{iso} = \mathbb{E}_{a,b\sim D}[d(\varphi(a * b), \varphi(a) \dot{*} \varphi(b))] = 0 \tag{41}$$

证明。

$$\sum_{a,b\in\mathbb{Q}} f(a, b).d(\varphi(a * b), \varphi(a) \dot{*} \varphi(b)) = 0 \tag{42}$$

对于所有 $a, b \in \mathbb{Q}$, $f(a, b) = f(a).f(b) > 0$。因此，

$$d(\varphi(a * b), \varphi_\Theta(a) \dot{*} \varphi(b)) = 0 \iff \tag{43}$$
$$\varphi(a) * \varphi(b) = \varphi(a * b) \tag{44}$$

现在，如果 $(\mathbb{Q}, *)$ 是一个群 （阿贝尔群），根据 (44) 和引理 A.1，$(\varphi(\mathbb{Q}) \dot{*})$ 是一个群（阿贝尔群 ）.

■

鉴于 $\mathbb{Q}$ 是可数的，可以在其上定义离散分布 (D)，其概率质量函数为 f，使得对于任意 $a \in \mathbb{Q}$， $f(a) > 0$。

定理 A.7。 函数 φ_{Θ_E} 是一一的，当对于某个度量 d： $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$,

$$\ell_{rec} = \mathbb{E}_{a\sim D}[d(\varphi^\dagger(\varphi(a)), a)] = 0 \tag{45}$$

证明。 根据期望定义，

$$\ell_{rec} = \sum_{a\in\mathbb{Q}} f(a)d(\varphi^\dagger(\varphi(a)), a) = 0 \tag{46}$$

根据假设，对于所有 $a \in \mathbb{Q}$， $f(a) > 0$。因此，

$$d(\varphi^\dagger(\varphi(a)), a) = 0 \iff \varphi^\dagger(\varphi(a)) = a \tag{47}$$

现在假设对于某个 a_1， $a_2 \in \mathbb{Q}$， $\varphi(a_1) = \varphi(a_2)$。

$$\varphi(a_1) = \varphi(a_2) \iff \tag{48}$$
$$\varphi^\dagger(\varphi(a_1)) = \varphi^\dagger(\varphi(a_2)) \iff \tag{49}$$
$$\mathrm{a}_1 = \mathrm{a}_2 \tag{50}$$

■

定理 A.8。 $<_\omega$ 是 φ上的神经序，当且仅当

$$\ell_{ord} = \mathbb{E}_{a,b\sim D}[[1(a < b) - 1(\varphi(a) <_\omega \varphi(b))]^2] = 0 \tag{51}$$

证明。 根据期望定义，

$$\ell_{ord} = \sum_{a,b\in\mathbb{Q}} f(a,b)[1(a<b) - 1(\varphi(a) <_\omega \varphi(b))]^2 = 0 \tag{52}$$

根据假设，对于所有 $a,\ b \in \mathbb{Q}$, $f(a,\ b) = f(a).f(b) > 0$。因此，

$$[1(a<b) - 1(\varphi(a) <_\omega \varphi(b))]^2 = 0 \iff \tag{53}$$
$$1(a<b) = 1(\varphi(a) <_\omega \varphi(b)) \iff \tag{54}$$
$$a < b \Leftrightarrow \varphi(a) <_\omega \varphi(b) \tag{55}$$

根据 (55) 和引理 A.3，关系 $<_\omega$ 是一个序。

■

定理 A.9。 代数结构 $(\varphi(\mathbb{Q})\ \dot{+}, \dot{\times}, <_\omega)$ 是一个有序域，并且同构于 $(\mathbb{Q}, +, \times, <)$ i f

$$\ell_{total} = \ell_{rec} + \ell_{iso} + \ell_{ord} = 0 \tag{56}$$

其中

$$\ell_{rec} = \mathbb{E}_{a\sim D}[d(\varphi^\dagger(\varphi(a)), a)] \tag{57}$$
$$\ell_{ord} = \mathbb{E}_{a,b\sim D}[(1(a<b) - 1(\varphi(a) <_\omega \varphi(b)))^2] \tag{58}$$
$$\ell_{iso} = \mathbb{E}_{a,b\sim D}[d(\varphi(a+b),\ \varphi(a) \dot{+} \varphi(b))] \tag{59}$$
$$+ \mathbb{E}_{a,b\sim D}[d(\varphi(a\times b),\ \varphi(a) \dot{\times} \varphi(b))] \tag{60}$$

证明。 该定理是定理 A.6、A.7 和 A.8 综合得出的结果。

■