Domain2Vec: Vectorizing Datasets to Find the Optimal Data Mixture without Training

Mozhi Zhang [1] Howe Tissue ✉ Lu Wang [2] Xipeng Qiu [1]

英寸

Abstract

我们介绍了 Domain2Vec ，这是一种新的方法，可以将任何数据集分解为几个元域的线性组合，这个新概念旨在捕捉数据集的关键底层特征。Domain2Vec 维护了一个元域词汇表，并使用分类器将任何给定的数据集分解为一个域向量，该向量代表了该词汇表上的分布。这些域向量在分布对齐假设（DA ^{2} ）下，为语言模型（LM）的预训练提供最佳数据混合选择，这种假设表明当训练集和验证集的数据分布更好地对齐时，可以实现较低的验证损失。此外，Domain2vec 可以无缝集成到先前的工作中，以建模域向量与 LM 性能之间的关系，大大提高了先前方法的效率和可扩展性。大量实验证明，Domain2Vec 帮助找到数据混合，以最小的计算开销提高下游任务性能。具体来说，Domain2Vec 在 Pile-CC 上仅使用原始 Pile 数据集混合所需计算量的 51.5% ，便达到了相同的验证损失。在等同的计算预算下，Domain2Vec 平均提高了下游性能 2.83% 。

1. 介绍

通过在大规模文本语料库上的训练，大型语言模型（LLMs）展现出其非凡的泛化能力（????）。LLMs 的训练数据集通常由来自不同来源的多个领域组成。最近的研究表明，这些领域的混合比例，称为数据混合，可以显著影响语言模型的效果（??），一个领域的数据可能会影响其他领域的性能（?）。通常，用于训练 LLMs 的数据混合是通过启发式方法或基于下游性能指标来确定的。然而，这些方法缺乏可扩展性，且常常产生次优的数据混合。因此，以一种可扩展且高效的方式识别最佳数据混合仍然是一个重要且具有挑战性的研究问题。最近，研究人员提出了多种方法来确定最佳的数据混合。第一类先前的工作通过选择来自不同领域或数据集的高质量数据来隐式地调整数据混合（???）。第二类工作则侧重于建模数据混合与大型语言模型 (LLMs) 性能之间的关系，并明确地调整不同数据集之间的数据混合（????????）。虽然先前的工作显示了很有前途的结果，但存在一些关键问题：1) 计算效率：例如，尽管 DoReMi 中的代理模型（?）只有 280 M 的参数，其估计的 FLOP 在计算仅 22 个数据集时就达到了 3.7×10^{19} 之高。此外，随着数据集数量的增加，这些方法的计算复杂度将非线性增长。2) 缺乏可扩展性：在建立数据混合与模型性能之间的函数关系后（??），如果数据集组成发生变化（例如，添加新数据集或筛选低质量数据等），之前拟合的函数无法直接应用。这需要重新采样新的数据混合、重新训练代理模型，并重新拟合函数，严重限制了这些方法的可扩展性。为了解决这些问题，我们引入了 Domain2Vec ，一个旨在矢量化数据集的新框架。这使我们能够在域向量空间中执行所有计算最佳混合比率的操作，当数据集发生变化时具有广泛的适用性。具体来说，Domain2Vec 维护一个元域的词汇表，我们假设任何数据集都可以被近似为几个元域以特定分布的线性组合。此分布可以作为给定数据集的向量表示（域向量）。

为了有效识别任何给定数据集的元域组成，我们建议使用元域分类器生成相应的域向量。在 Domain2Vec 的基础上，我们引入分布对齐假设（DA ^{2} ），以寻找 LM 预训练的最佳数据混合。DA ^{2} 指出，当训练数据集的域向量与验证数据集的域向量更好地对齐时，可以实现较低的验证损失。基于 DA ^{2} ，我们可以轻松找到无需训练的最佳数据混合。

此外，Domain2Vec 可以无缝集成到先前的工作中，如 RegMix（?）。与先前的方法不同，这些方法建模数据混合与语言模型性能之间的关系（??），我们建模了 Domain2Vec 提供的领域向量与模型性能之间的关系，进一步增强了先前工作的效率和可扩展性。

总而言之，我们强调我们的贡献如下：

1. 我们引入 Domain2Vec 来矢量化数据集，并提出将数据集视为元领域的组合。我们展示了一种使用元领域分类器来矢量化数据集的高效流程。

2. 我们提出了一种分布对齐假设（DA ^{2} ），这是一种无需训练的方法，用于识别最佳数据混合。我们进一步展示了如何将 Domain2Vec 无缝集成到之前的工作中，以提高效率和可扩展性。

[1] 上海，中国，复旦大学计算机科学学院 [2] Ritzz-AI . Correspondence to: Howe Tissue (project lead) < h-sun20@tsinghua.org.cn >.

Proceedings of the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

3. 我们验证了 Domain2Vec +DA [2] 和 +RegMix 在文本生成和下游任务中的有效性。实验结果表明，我们的方法能够在不训练代理模型的情况下准确预测各种数据混合的性能。此外，我们可以识别出仅使用其 0.26 % 计算成本的数据混合，达到与 DoReMi (?) 相当的下游性能。

在本节中，我们介绍了 Domain2Vec ，这是一种算法，可以将数据集分解为各种元域的线性组合，并允许我们通过一个标准化向量表示数据集的底层特征。我们还概述了构建 Domain2Vec 词汇和训练元域分类器的流程。

关键假设。 Domain2Vec 保持一个词汇，一个元领域的集合。假设我们有 n 个元领域 $\mathcal{D}_j^*$ ($1 \leq j \leq n$) ，其中 $\mathcal{D}_j^*$ 表示为 $\boldsymbol{e}_j$ ，这是一个独热向量，其中只有第 j 个元素为 1 。我们假设，对于任何给定的数据集 $\mathcal{D}$ ，它可以通过这些元领域的线性组合表示为一个领域向量 $\boldsymbol{v}$ 。具体来说，

$$\boldsymbol{v} \approx \sum_{j=1}^{n} v_j \cdot \boldsymbol{e}_j, \tag{1}$$

其中 $\boldsymbol{v}$ 的每个元素 v_j 表示数据集 $\mathcal{D}$ 在 $\mathcal{D}_j^*$ 上的投影(权重)。因此，$\boldsymbol{v} = [v_1, v_2, ..., v_n]^\top$ 可以是数据集 $\mathcal{D}$ 在元领域上的一种表示（分布）。然而，构建这些元领域的理想方法仍需要被确立。接下来，我们将介绍如何从大规模无标签文本语料库中构建元领域。

根据上述关键假设，我们将元域定义为一组实际数据集，它们在域矢量空间中作为基础，允许通过这些具体数据集的线性组合来表示该空间中的任何未知域。这些构建的元域可代表来自任何来源的数据集，应该满足以下三个属性，类似于线性代数中基的属性：

1. 生成集。构成元域的各个域应尽可能多样化和全面。
2. 线性独立性。构建的这些元域之间应该有明显的区别。
3. 计算效率（可选）。用于构建元域的方法应具有计算效率。

为了获得多样化和全面的元域，我们从超过 100 个粗粒度的英文、中文 [1] 和代码来源中收集数据。经过去重，我们获得了约 5.2 TB 的文本数据，包括超过 1 亿个文档。大型语料库与标准的大规模 LLM 预训练有相似的来源组成，包括公共爬取（CC）、维基百科、社交媒体平台、arXiv、代码、新闻、书籍等。可以假设，这些语料库已经包含了尽可能多样化和全面的内容，以满足“跨越集合” [2] 的要求。

[1]在本文中，我们主要针对英语和中文。

[2]由于去重预处理和语料库之间的本质差异，“线性独立”的要求也自然得到了满足。

在获取语料库后，我们的目标是提取语料库中的元领域，即将语料库划分为一些（语义上）不同的聚类，用作元领域。我们采用 k -means (??) 聚类算法来实现分离，并分别利用 bge-small-en-v1.5 和 bge-small-zh-v1.5 (?) 计算英文和中文文档的嵌入。参见图 2 以了解元领域数量与惯性（衡量每个数据点与其中心的距离）之间的关系。此外，我们直接根据编程语言划分代码数据。最终，我们构建了 260 （120 中文 + 120 英文 + 20 代码）个独特的元领域。语料库中的每个文档都标记了它来源于哪个元领域。

元域分类器 现在，我们提出一种使用先前建立的元域来表示未见数据集的方法。该方法简单却有效：我们将未见数据集中的每个文档分配给其对应的元域，然后计算所有文档的整体分布。这种全面的表示法捕捉整个数据集的总体领域特征。形式上，假设存在一个元域分类器，对于任何给定文档 $doc \in \mathcal{D}$ ，p_i 表示 doc 属于第 i 个元域的概率，其中 $\|\boldsymbol{p}\|_1 = 1$ 。对于未见数据集 $\mathcal{D}$ ，我们可以抽样 N 文档 [3] 然后对这些样本的领域向量取平均值。形式上，数据集 $\mathcal{D}$ 的领域向量 $\boldsymbol{v}$ 为，

$$\boldsymbol{v} \approx \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{p}_i, \tag{2}$$

然后，我们可以使用向量 $\boldsymbol{v}$ 来近似表示任何未知数据集 $\mathcal{D}$ 的特征。同时，在 LLMs 的预训练阶段，我们通常有来自许多来源的训练数据集 $\mathcal{D}_{train} = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_m\}$ 。我们可以根据公式 ?? 和 2 将每个数据集转换为领域向量。因此，$\mathcal{D}_{train}$ 可以近似表示为 $\boldsymbol{V}_{train} = [\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_m]$ ，其中 $\boldsymbol{V}_{train} \in \mathbb{R}^{n \times m}$ 和 n 是元领域的数量。

具体来说，我们训练了一个 260 类别的分类器，以确定给定文档来源哪个元域。我们对 Qwen2-1.5b-base (?) 进行微调，以平衡准确性和效率。训练后，元域分类器在我们构建的测试集上获得了 74.73% 的分类准确率。为了进一步评估元域分类器的性能，我们还从 The Pile (?) 的每个子数据集中抽取了 1,000 个样本。根据方程 2 ，我们为每个子数据集获得了由元域分类器预测的域向量，如图 1 所示。The Pile 的子数据集在元域上的分布表现出不同的模式。这种现象不仅表明不同的元域具有显著的语义差异，同时也表明我们的分类器可以准确区分不同未见数据集的语义特征。

在本节中，我们首先介绍最优数据混合发现的任务形式。然后，我们展示了如何使用 Domain2Vec 在不需要额外训练的情况下识别最优数据混合的方法。我们介绍了两种方法：第一种方法基于分布对齐假设（DA [2] ）。此外，我们展示了如何将我们的 Domain2Vec 与之前建模混合比例和最终性能关系的工作相结合，显著增强这些现有方法的可扩展性。

在 LLMs 的预训练阶段，我们通常从 m 个来源（例

[3]在本文中，我们设置了 $N = 1000$ ，这足以获得一个准确且稳定的领域向量。

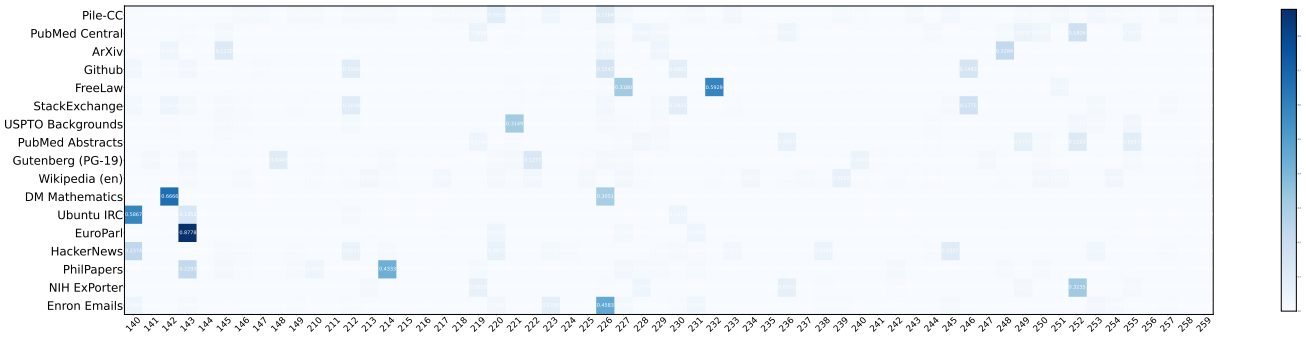

Figure 1. (?) 的每个子数据集的域向量，其中每一行对应一个子数据集，每一列对应一个元域。属于特定元域的数据比例越高，相应单元格的颜色越接近 蓝色 。为了清楚起见，我们展示了一些英语元域上的分布。完整的图示见图 7 。

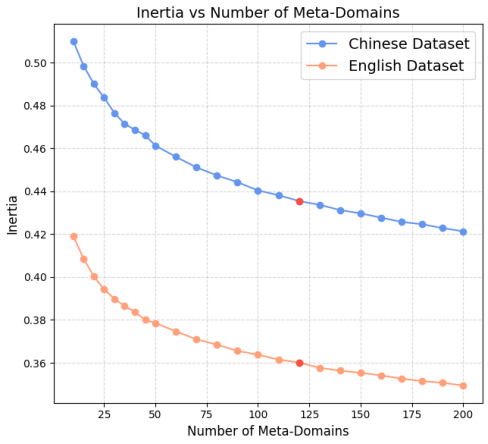

Figure 2. 元域的数量与惯性。

如，arXiv、维基百科等）收集训练数据集 $\mathcal{D}_{train} = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_m\}$ 。我们还预先定义了一个验证集 $\mathcal{D}_{valid}$ ，该验证集具有高质量并指示最终性能。注意，$\mathcal{D}_{valid}$ 通常与 $\mathcal{D}_{train}$ 独立且同分布。例如，? 采用 Pile-CC (?) 作为验证集，而 ? 采用 LIMA (?) 作为验证集。因此，数据混合 $\boldsymbol{r} = [r_1, r_2, ..., r_m]^\top$ ，其中 $0 \leq r_i \leq 1$ 和 $\sum_{i=1}^{m} r_i = 1$ ，指定了 m 数据集的混合比例。将训练后的语言模型记为 θ ，语言模型的验证损失记为 $\mathcal{L}_\theta$ 。寻找最佳数据混合 $\boldsymbol{r}^*$ 的目标通常是最小化验证损失，如公式 ?? 中正式所示。我们将 $\mathcal{L}^{\mathcal{D}_{valid}}(\boldsymbol{r})$ 记为在数据混合 $\boldsymbol{r}$ 上预训练的语言模型的验证损失。

1.1. 初步研究：混合比率排名在不同模型规模中保持一致

我们首先针对一个关键的研究问题进行试点研究：最优数据混合能否在不同的模型规模上泛化？如果答案是肯定的，那么就有可能通过简单地训练一个小的代理模型，甚至更高效地，无需训练任何模型，来确定最优混合比例。为了回答这些问题，我们在表中混合了 C4 (?) 和知识堆 (?) 的不同数据混合 $(0, 0.2, \cdots 1.0$)。我们使用标准的语言模型损失从零开始预训练两个参数量分别为 83 M 和 1.6 B 的语言模型。在预训练过程中，我们评估了使用不同混合比例在 The Pile (?) 和 RedPajama (?) 的 20 子集上训练的模型的验证损失，如图 3 所示。更多验证集的结果可以在图 8 和 9 中看到。有两个发现：

- 每个验证集都有一个最佳混合比率，不同验证集之间的排名差异显著。
- 对于相同的验证集，在不同模型规模下，数据混合比例的排名几乎保持不变。我们计算了 83 M 模型和 1.6 B 模型在不同验证集上的数据混合排名的相关系数。分析得出一个 Spearman 系数为 0.9743 和一个 Pearson 系数为 0.9947 ，这为这种一致性提供了强有力的统计证据。这些异常高的相关值强烈支持我们的发现，即在相同的验证基准上评估时，最佳混合比例在很大程度上与模型规模无关。

这些发现与 ? 的先前工作一致，这表明可以在不进行训练的情况下（见章节 ?? ）或仅仅进行小模型训练的情况下找到最优数据混合（见章节 1.2 ）。

我们介绍了如何直接应用我们提出的 Domain2Vec 来寻找最优的数据混合。我们注意到，当训练集的数据分布与给定的验证集更好地对齐时，会达到较低的验证损失 $\mathcal{L}^{\mathcal{D}_{valid}}$ 。这一现象的一个基本问题是我们如何对各种数据集的数据分布进行建模？幸运的是，根据第 ?? 节，对于训练数据集 $\mathcal{D}_{train}$ ，我们获得了表示 $\mathcal{D}_{train}$ 语义特征的向量表示 $\boldsymbol{V}_{train} \in \mathbb{R}^{n \times m}$ 。相应地，对于验证集 $\mathcal{D}_{valid}$ ，我们也有其向量表示 $\boldsymbol{v}_{valid}$ 。在与数据混合 $\boldsymbol{r}$ 混合后，$\mathcal{D}_{train}$ 在元域上的最终分布为 $\boldsymbol{v}_{train} = \boldsymbol{V}_{train} \cdot \boldsymbol{r}$ 。因此，基于分布对齐假设，方程 ?? 可以等效地写为：其中，Dist$(\cdot, \cdot)$ 是用于测量两个向量之间相似性的距离函数。从理论上讲，有许多距离函数的选项，包括 Wasserstein（最优传输）距离，欧几里得距离等。在本文中，我们使用向量之间的 Huber 损失 (??) 来测量距离。我们在附录 ?? 中讨论了不同距离函数的选择。我们在附录 ?? 中展示了 Domain2Vec +DA [2] 的伪代码。

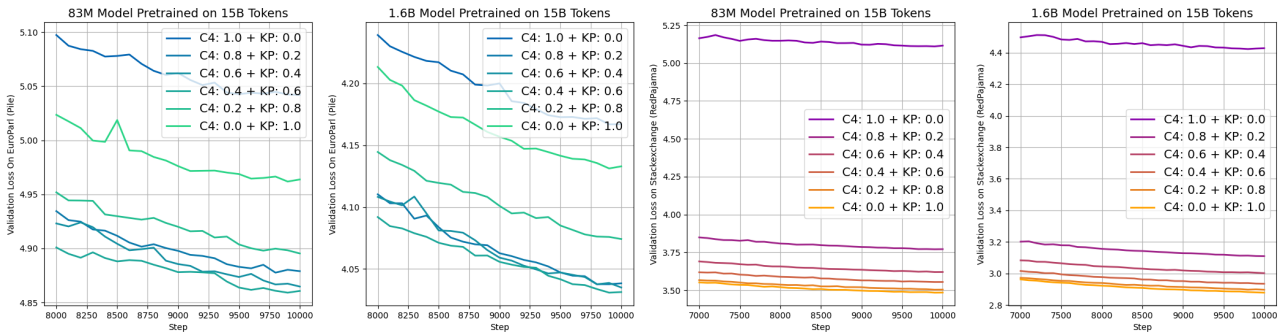

Figure 3. 使用表格 ?? 中的数据混合训练的模型在 EuroParl（The Pile）和 Stackexchange（RedPajama）上的验证损失。其他验证集上的验证损失在附录 ?? 中显示。

1.2. 将 Domain2Vec 应用于早期工作

有一条典型的研究路线是专注于确定最佳混合比例，其目标是通过各种功能方法建模这些比例与最终验证损失之间的关系。也就是说，这些方法模型化 $\mathcal{L}^{\mathcal{D}_{valid}}(\boldsymbol{r}) = f(\boldsymbol{r})$ ，其中 $f(\cdot)$ 可以采用先前工作中提出的各种合理形式。例如，

- 数据混合律 (?) 采用 $f(\boldsymbol{r}) = c_i + k_i \cdot \exp\left(\sum_{j=1}^{m} t_{ij} \cdot r_j\right)$ 来预测训练域 i 上的验证损失，其中 c_i, k_i, t_{ij} 是所有未确定的待拟合参数。

- RegMix (?) 最初采用线性回归方法，将验证损失建模为 $f(\boldsymbol{r}) = \boldsymbol{w}^\top \boldsymbol{r}$ 其中 $\boldsymbol{w}$ 需要拟合。此外，它通过使用 LightGBM (?) 来更有效地拟合函数 $f(\cdot)$ 来推进这一概念。

我们可以直接将 Domain2Vec 与这些方法集成，而无需修改它们的核心功能，而是在域向量空间中执行计算。因此，我们解决了这些方法的两个固有限制：(1) 效率：对于有 m 个变量的 $f(\cdot)$ 建模，预计需要针对不同的 $\boldsymbol{r}$ 运行 $O(m^2)$ 次实验以收集拟合点；(2) 可扩展性：当引入一个新的训练源时，必须重新收集拟合点并重新拟合 $f(\cdot)$ ，这缺乏可扩展性。

具体而言，我们创新地建立了验证损失 $f_i(\cdot)$ 与混合训练数据集比例 $\boldsymbol{r}$ 之后的域向量 $\boldsymbol{v}_{train}$ 之间的关系，这里的验证损失是在第 i 个元域 $\mathcal{D}_i^*$ 上的，记作 $\mathcal{L}^{\mathcal{D}_i^*}$ 。正式地，对于每个元域，我们有

$$\mathcal{L}^{\mathcal{D}_i^*}(\boldsymbol{r}) = f_i(\boldsymbol{v}_{train}) = f_i(\boldsymbol{V}_{train} \cdot \boldsymbol{r}), 1 \le i \le n. \quad (3)$$

。方程 3 使得在给定数据混合的情况下，可以预测任何元域上的验证损失，这也是我们要拟合的函数。对于看不见的验证数据集，回忆一下任何包含 $\mathcal{D}_{valid}$ 的数据集也可以被视作元域的线性相加，$\mathcal{D}_{valid}$ 的域向量记作 $\boldsymbol{v}_{valid} = [q_1, q_2, \cdots, q_n]^\top$ 。因此，我们有

$$\begin{aligned}\mathcal{L}^{\mathcal{D}_{valid}}(\boldsymbol{r}) &= \sum_{i=1}^{n} q_i \cdot \mathcal{L}^{\mathcal{D}_i^*}(\boldsymbol{r}) = \sum_{i=1}^{n} q_i \cdot f_i(\boldsymbol{v}_{train}) \\ &= \sum_{i=1}^{n} q_i \cdot f_i(\boldsymbol{V}_{train} \cdot \boldsymbol{r}).\end{aligned} \quad (4)$$

。现在，我们通过所提出的 Domain2Vec 在域向量空间中将验证损失与混合比例连接起来。通过最小化 $\mathcal{L}^{\mathcal{D}_{valid}}(\boldsymbol{r})$ ，可以可行地搜索出最佳混合比例 $r^\star$ 。注意，这种连接只建立在元域之上（即 $1 \le i \le n$ 的 f_i ），并且可以无成本地适应：(1) 任何看不见的验证集；(2) 任何看不见的训练集；(3) 任何数量的训练集。得益于这一特性，我们可以通过 Domain2Vec 在先前的方法中实现效率和扩展性。

根据 RegMix (?) 的方法，我们使用 LightGBM (?) 作为 $f(\cdot)$ 来拟合每个元域的方程 3 （称为 Domain2Vec +RegMix）。Domain2Vec + RegMix 的伪代码展示在附录 ?? 。我们从 Dirichlet 分布中抽取 $10,500$ 个不同的混合比例，并通过训练 $10,500$ 个小 LM 获取每个元域的验证损失。我们还保留一些混合比例作为测试集，并进行实验以评估拟合函数 $f(\cdot)$ 是否能够准确预测未见混合比例的验证损失。对于测试集中的各种混合比例，我们使用 Spearman 系数来衡量预测排序与未见混合比例下性能实际排序之间的相关性。注意，我们采用相关系数是因为它是一个比平均损失误差更一般的指标，目标是找到比其他更好的混合比例。此外，初步研究表明，预测排序在不同模型规模间保持稳定，而预测损失在不一致的模型规模下则变得无意义。如图 4 所示，Spearman 系数随着我们用于训练和收集拟合点的混合比例数量的增加而上升。最终，我们获得了一个超过 90% 的 Spearman 系数，这对于预测各种元域的良好混合比例非常准确。

在本节中，我们描述了实现和结果，以展示如何通过降低计算成本来识别最佳数据混合。优化数据混合的目标是提高语言模型的性能。语言模型的性能可以从两个角度进行评估：1）文本生成，通常通过验证集上

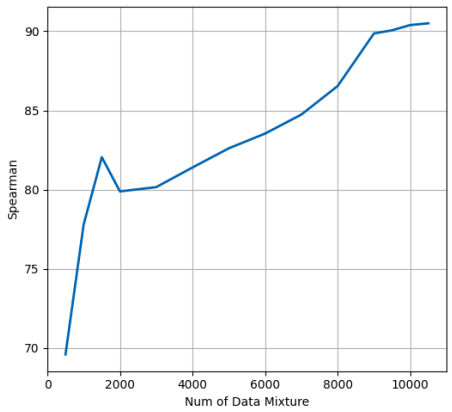

Figure 4. 训练数据混合数量与斯皮尔曼相关系数之间的关系。

的语言模型损失来衡量。我们的目标是通过找到最佳数据混合来最小化验证损失；2）下游任务性能。我们的目标是优化任务性能。实验结果的概览，通过应用 Domain2Vec ，我们可以在各种设置（例如训练集和验证集）下准确预测数据混合的排名。我们还实现了与 The Pile 的原始数据混合相当的验证损失，同时仅使用了 51.5 % 的计算资源。此外，我们仅使用 0.26% 的计算成本，找到一种性能可与 DoReMi 等强基准相媲美的数据混合。

1.3. 验证损失最小化

我们设计了一些训练和验证数据集来评估我们方法的性能，以最小化验证损失。我们的训练数据集包括 C4 和 Knowledge Pile。C4 是一个很大且清理过的 Common Crawl 语料库版本。Knowledge Pile 是一个高质量的数据集，显著提高了 LLM 在知识相关和数学推理任务中的性能。我们在各种验证数据集上进行实验，以进行全面的评估。我们从 The Pile 和 RedPajama 中选择验证数据集。由于验证数据集之间的最佳混合比率不同，我们预测在不同预设混合比率下的性能排名。具体而言，我们将 C4 和 Knowledge Pile 以不同数据混合方式作为训练集进行混合，如表所示。

我们从头开始使用标准语言模型损失预训练具有 83 百万和 1.6 十亿参数的 LLaMA-like 型号 (?) 。两个模型的批量大小为 1.5 百万个标记，最大序列长度为 4,096 。我们使用 AdamW 优化器 (?) ，并将梯度裁剪在 1.0 。学习率在前 100 步线性上升至 2e-4，然后在 10,000 步内使用余弦调度器衰减至 2e-5。更多参数详细信息见表 5 。然后，我们使用 Spearman 和 Pearson 相关系数评估 Domain2Vec ，考察预测排序与实际排序之间的相关性。我们将 Domain2Vec 与随机排序和一个基于嵌入的基线（记为 k NN）进行比较。具体而言，我们使用 bge-small-v1.5 获取嵌入，并应用均值池化来为每个数据集和元域生成唯一的嵌入。随后，我们基于欧几里得距离应用 k NN 计算来自每个元域的训练和测试数据集的概率分布，将这些分布视为新的域向量。

实验结果。 我们在图 3 和附录 ?? 展示了针对各种数据组合的验证损失曲线。可以观察到，在大多数验证集上，结合使用知识堆中的数据显著降低了验证损失。我们应用了第 ?? 节中描述的两种基于 Domain2Vec 的方法来对表 ?? 中的数据组合进行排序。

如表 1 所示，Domain2Vec 预测的排名与实际排名表现出很强的正相关性，显著优于随机猜测和 k NN。k NN 方法的有效性部分验证了我们元域词汇构建背后的合理性。还需要注意的是，我们的方法是一种无训练方法，与之前依赖训练小代理模型来对数据混合进行排名的研究不同。尽管设置更加具有挑战性，我们的方法仍可以准确预测不同数据混合的排名。

Table 1. 部署 Domain2Vec 以预测不同验证集排名的结果。

Metrics	Random	k NN	Domain2Vec+DA [2]	Domain2Vec+RegMix
Pearson	0.0300	0.4014	0.5833	0.3881
Spearman	0.0497	0.3543	0.6657	0.4629

在本节中，我们演示了如何使用 Domain2Vec 来识别最大化下游任务性能的最佳数据混合。一个挑战是建模数据混合与下游性能之间的关系。幸运的是，? 发现 Pile-CC 的验证损失与他们的评估中的下游性能最相关。为了与先前的工作保持一致，我们遵循并使用与 ? 相同的验证数据集。因此，我们的代理目标是识别一种在 Pile-CC 上最小化验证损失的数据混合。实验结果表明，Domain2Vec 预测了一种与 DoReMi (?) 性能相当的数据混合，同时只使用 0.26% 的计算成本。

我们使用 RegMix (?) 并选择 The Pile (?) 作为我们的训练数据集。The Pile 是一个用于 LLM 预训练的 825 GB 英语文本语料库。根据 RegMix 的标准，我们只使用 The Pile 的 17 个没有版权问题的组成部分。我们的目标是识别能够在 Pile-CC 子集上最小化验证损失的数据混合，以改善下游任务的性能。我们将我们的方法与几个基准进行比较，包括 Human（原始数据混合）、DoReMi (?) 和 RegMix (?) 。包含 Pile-CC Only 基准（仅在 Pile-CC 子集上训练模型）以验证 Pile-CC 验证损失与下游性能之间的强相关性。每个基准的数据混合在表 3 中展示。

我们从头开始预训练类似 LLaMA 的 (?) 模型，使用标准语言建模损失，模型大小范围从 106 M 到 1 B 参数。根据 ? ，每个模型的 token 数量是对应参数大小的 20 倍。所有模型采用 1 M tokens 的批量大小和最大的序列长度 4,096 。我们应用 AdamW (?) 优化器，并在 1.0 处进行梯度裁剪。学习率在 1,000 步内线性升温至 6e-4，然后在训练结束时使用余弦调度器衰减至 0 。更多参数详见表格 5 。为了评估，我们追踪不

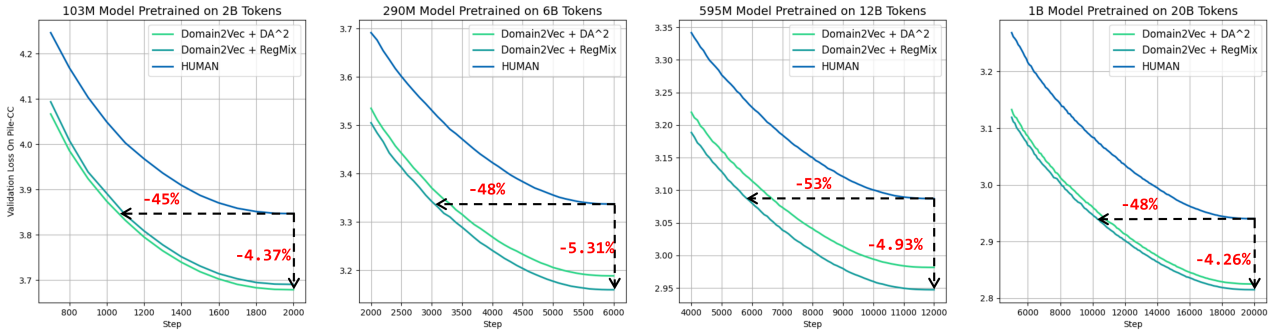

Figure 5. 在 Pile-CC 子集上的验证损失。Domain2Vec 实现了与 Human（使用 The Pile 中的原始数据混合进行训练的模型）相当的验证损失，但仅用了 Human 的 51.5% 训练计算成本。在相同的训练成本下，Domain2Vec 可以将验证损失相比于 Human 减少约 4.72% 。

同模型大小在 Pile-CC 验证损失上的表现。此外，我们使用以下基准评估不同数据混合的表现：Social IQA (?) ，HellaSwag (?) ，PiQA (?) ，OpenBookQA (?) ，Lambada (?) ，SciQ (?) ，ARC Easy (?) ，COPA (?) ，RACE (?) ，LogiQA (?) ，WinoGrande (?) ，和 MultiRC (?) 。我们利用 LM Evaluation Harness (?) 评估这些模型，并在表格 ?? 中报告 0-shot 到 5-shot 设置下的平均分数。

我们通过应用方程 ?? （Domain2Vec +DA ^{2} ）和方程 4 （Domain2Vec +RegMix) 来预测最佳数据组合。我们基于这些组件的标记分布，从 Dirichlet 分布生成 100,000 个数据组合。利用这些组合，我们通过提出的两种方法预测最佳数据组合。我们选择预测的前 100 个数据组合并将其平均作为最终的数据组合。这个技巧与之前的工作 (?) 一致，以获得更准确和稳定的结果。作为标准做法，The Pile 的每个子集最多训练一个 epoch。在优化组合比例 $\boldsymbol{r} = [r_1, r_2, \cdots, r_m]^\top$ 时，除了常见的限制 $0 \leq r_{1\cdots m} \leq 1$ 和 $\sum_{i=1}^{m} r_i = 1$ 外，注意还有一个数据量限制，即 $\#TotalTokens \cdot r_i \leq |\mathcal{D}_i|$ ，这是为了去除在某些子集中需要超出标记的数据组合。因此，Domain2Vec 预测的最佳数据组合可能会因训练的标记数量和模型的大小而异。这个限制不同于第 1.3 节，其中每个数据集的大小被视为无限。

如图 5 所示，我们提出的 Domain2Vec + DA ^{2} 和 Domain2Vec + RegMix 在 Pile-CC 上的训练效率相比于 Human 有显著提高。具体来说，Domain2Vec + DA ^{2} 和 Domain2Vec + RegMix 分别只需大约 55.38% 和 51.50% 个训练步骤即可达到与 Human 相同的验证损失。在相同计算预算下，相较于 Human，Domain2Vec + DA ^{2} 和 Domain2Vec + RegMix 分别将验证损失降低了约 4.04 % 和 4.64 % ，并分别平均提高了下游性能 1.89% 和 2.83% 。在表 ?? 中，我们报告了在各基线数据混合上训练的语言模型在一系列下游任务中的平均性能。“仅使用 Pile-CC” 的平均准确率比 Human 提高了 3.54 % ，这表明在 Pile-CC 上训练更多的 tokens 可以提升下游性能。重要的是，当我们将 Pile-CC 视为验证集时，“仅使用 Pile-CC” 表现良好。然而，在更

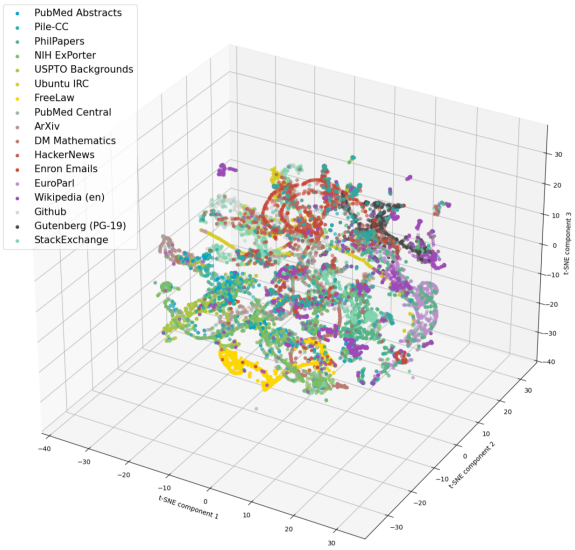

Figure 6. 《The Pile》中域向量的可视化（t-SNE）。

实际的场景中，验证集并非如此，我们无法手动找到这样一个可以对应验证集的黄金训练集。为此，我们可以使用我们提出的 Domain2Vec，通过混合来自不同来源的数据集，以最低的成本获得可比较的下游性能。值得注意的是，Domain2Vec + DA ^{2} 和 Domain2Vec + RegMix 只使用了 DoReMi 所需的 FLOPs 的大约 0.26 % ，便取得了与 DoReMi、RegMix 相当的性能，这证明了 Domain2Vec 的计算效率。

为了进一步研究，我们使用 t-SNE (?) 来可视化 The Pile 中每个组成部分的领域向量，如图 6 所示。这种可视化揭示了学习向量的几个理想属性。表示空间表现出强烈的聚类行为，其中语义相关的数据集自然聚集在一起，这表明有效捕捉了领域特定的特征。相关领域如学术文献 (PubMed, arXiv) 和技术库 (GitHub, StackExchange) 展现出空间上的一致性，同时在不同领域之间保持明确但灵活的边界。表示涵盖了 The Pile 中的不同领域，展示了在异构数据类型中强大的

泛化能力。

1.4. 关于过拟合的讨论

我们注意到，一些读者将我们的方法解释为一种“过拟合”：在选定的验证集上优化。我们提供以下解释：

最近关于优化数据混合的研究可以大致分为两个方面。第一个方面是通过基于数据质量从各种数据集进行降采样来隐式调整数据混合。例如，？ 提出了 RHO-1，该方法使用选择性语言模型来选择与理想比例一致的数据混合的标记。与标记级选择不同的是，？ 使用小型参考模型的困惑度来过滤低质量样本。？ 展示了影响分数可以指导数据重加权，而他们随后的工作则介绍了一种消除了对参考模型需求的在线数据选择方法。

第二条关注通过建模数据混合与语言模型性能之间的关系来显式调整数据混合。最简单的方法是观察各种数据混合的性能并选择最佳的一个，如在 Gopher 训练中所做的那样 (?) 。这是昂贵的，对于较大的模型难以扩大规模。 ？ 提出 DoReMi，使用一个小的代理模型对来自不同领域的数据进行重新加权，提高较大模型的训练效率。然而，DoReMi 仍需要一个预训练的参考模型，这增加了计算成本，并使得定义一个理想的参考模型变得困难。一些工作旨在建模数据混合与语言模型性能之间的函数关系。受到扩展定律的启发 (??) ， ？ 引入了数据混合定律，该定律使用指数形式描述这种关系。 ？ 提出了 BiMix，这是一种同时考虑计算和数据混合的扩展定律。 ？ 和 ？ 开发了用于持续预训练的扩展定律，并建模了混合比作为一个变量如何影响语言模型损失。最近， ？ 提出了线性回归来建模不同数据混合下的验证损失，展示了强大且有前景的性能。

所有这些先前的工作都面临两个主要问题：1) 计算效率：例如，当应用于大约 20 个数据集时，DoReMi 和 RegMix 的估计 FLOPs 分别达到 3.7×10^{19} 和 3.5×10^{18} 。此外，随着数据集数量的增加，这些方法的计算复杂性将非线性增长。2) 缺乏可扩展性：当训练数据集的组件发生变化时（例如，添加一些新数据集），以前的方法 (??) 需要重新采样数据混合、重新训练代理模型，然后重新执行拟合过程。在本文中，我们引入了 Domain2Vec ，以将任何数据集分解为元域的线性组合。Domain2Vec 与先前的元学习工作（例如 ？ 和 ？ ）共享一些概念，这些工作在潜在空间中探索数据集表示。虽然共享这个概念，但 Domain2Vec 在目的和实现上都不同，并且我们专注于 LM 预训练中的数据混合。在这项工作中，我们引入了一种新方法 Domain2Vec ，通过将数据集分解为多个元域的线性组合来捕获数据集的潜在特征。这使我们能够获得任意数据集的向量化表示。在这些域向量的基础上，我们通过分布对齐假设（DA ^{2} ）引入了一种无训练的方法，以识别语言模型预训练的最佳数据混合。此外，Domain2Vec 无缝集成到现有方法中，通过在不需要在训练数据集变化时重新训练的情况下，建立模型性能与域向量之间的直接关系，大大提高了方法的效率和可扩展性。我们的实验结果表明，Domain2Vec +DA ^{2} 和 Domain2Vec +RegMix 在较低的计算开销下，与现有方法相比，能够实现相当的文本生成和下游任务性能。我们相信这项工作为优化语言模型预训练的数据混合提供了宝贵的见解，并为更高效的训练策略铺平了道路。

这篇论文的目标是推进机器学习领域的发展。我们的工作可能会带来许多潜在的社会影响，但没有哪一点是我们认为必须在此特别强调的。

本工作得到中国国家自然科学基金（编号：U24B20181）和福建省自然科学基金（编号：2024J08371）的资助。

在本节中，我们详细描述了语言模型预训练的分布对齐假设。

在寻找语言模型预训练的最优数据混合的场景中，验证集 $\mathcal{D}_{valid}$ 是固定的，我们旨在调整数据混合来构建训练集 $\mathcal{D}_{train}$ ，以通过公式 ?? 计算得到更低的验证损失，其中 $\hat{\theta}$ 是预训练语言模型的参数。通常，我们通过下一个标记预测 (?) 类似公式 ?? 来预训练语言模型。也就是说，我们需要找到一个 $\hat{\theta}$ 来最大化 $X \sim \mathcal{D}_{train}$ 的期望概率，这也被称为最大似然估计 (MLE)。当 $\mathcal{D}_{\text{train}}$ 和 $\mathcal{D}_{\text{valid}}$ 的数据分布对齐时，语言模型预训练过程的优化目标等于找到一个 $\hat{\theta}$ 来最大化 $X \sim \mathcal{D}_{valid}$ 的期望概率。因此，我们引入了语言模型预训练的分布对齐假设，这是一种无需训练即可找到最优数据混合的新方法。

在算法 1 中，我们给出了用于获取训练和验证数据集的领域向量的伪代码。

在算法 2 和 3 中，我们展示了如何使用 Domain2Vec 来寻找最佳数据混合的伪代码，包括分布对齐假设，并将 Domain2Vec 应用于 RegMix。

请注意，当应用 Domain2Vec +DA ^{2} 或 Domain2Vec +RegMix 时，为了获得更稳定和准确的结果，可以对 K 个采样候选数据混合中的 k -最佳比率进行平均。在伪代码中，我们以 top- 1 作为一个例子。在第 1.3 节中，我们采用 top- 1 进行直接比较，而在第 ?? 节中，我们采用与 RegMix (?) 一致的 top- 100 。

Algorithm 1 Domain2Vec

Require: Training datasets $\mathcal{D}_{train} = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_m\}$, validation dataset $\mathcal{D}_{valid}$, meta-domain classifier Classifier

Domain vectors $V_{train} = []$
for $i = 1$ to m do
  Sample N documents from $\mathcal{D}_i$
  $\boldsymbol{v}_i = \frac{1}{N} \sum_{j=1}^{N} \text{Classifier}(doc_j)$, where $doc_j \in \mathcal{D}_i$
end for
Sample N documents from $\mathcal{D}_{valid}$
$\boldsymbol{v}_{valid} = \frac{1}{N} \sum_{j=1}^{N} \text{Classifier}(doc_j)$, where $doc_j \in \mathcal{D}_{valid}$
Return: $\boldsymbol{V}_{train} = [\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_m], \boldsymbol{v}_{valid}$

Algorithm 2 Domain2Vec +DA [2]

Require: Domain vectors of training datasets $\boldsymbol{V}_{train} = [\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_m]$, domain vectors of validation dataset $\boldsymbol{v}_{valid}$, token distribution of training datasets $\boldsymbol{a}_{train}$.

Sample K candidates data mixture $\boldsymbol{r}_i$ from Dirichlet($\boldsymbol{a}_{train}$)
The optimal data mixture $\boldsymbol{r}^* = \boldsymbol{r}_1$
for $i = 2$ to K do
  if Dist($\boldsymbol{V}_{train} \cdot \boldsymbol{r}, \boldsymbol{v}_{valid}$) < Dist($\boldsymbol{V}_{train} \cdot \boldsymbol{r}^*, \boldsymbol{v}_{valid}$) then
    $\boldsymbol{r}^* = \boldsymbol{r}_i$
  end if
end for
Return: the optimal data mixture $\boldsymbol{r}^*$

Algorithm 3 Domain2Vec + 正则化混合

Require: Domain vectors of training datasets $\boldsymbol{V}_{train} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_m]$, domain vectors of validation dataset $\boldsymbol{v}_{valid} = [q_1, q_2, \cdots, q_n]^\top$, token distribution of training datasets $\boldsymbol{a}_{train}$, fitted model for each meta-domain $f_i(\cdot)$.

Sample K candidates data mixture $\boldsymbol{r}_i$ from Dirichlet($\boldsymbol{a}_{train}$)
The optimal data mixture $r^* = r_1$
Def $\mathcal{L}(\boldsymbol{r}) = \sum_{i=1}^{n} q_i \cdot f_i(\boldsymbol{V}_{train} \cdot \boldsymbol{r})$
for $i = 2$ to K do
  if $\mathcal{L}(\boldsymbol{r}_i) < \mathcal{L}(\boldsymbol{r}^*)$ then
    $\boldsymbol{r}^* = \boldsymbol{r}_i$
    $\mathcal{L}(\boldsymbol{r}^*) = \mathcal{L}(\boldsymbol{r}_i)$
  end if
end for
Return: the optimal data mixture $\boldsymbol{r}^*$

在本节中，我们将展示在 The Pile 上使用的不同方法的数据混合，这些方法用于本论文中的再现。在表 3 中，我们展示了由 Domain2Vec + DA 和 Domain2Vec + RegMix 预测的最佳数据混合。需要注意的是，为了避免过拟合问题，The Pile 的任何子集最多只会训练一个 epoch。因为我们采用拒绝采样来过滤掉某些不合理的数据混合。随着模型大小的变化，预测的数据混合可能会发生变化。

在本节中，我们报告了在 arXiv、C4、Book3、PG19 从 RedPajama (?) 和 BookCorpus2、DM Mathematics、Enron Emails、FreeLaw、HackerNews、NIH ExPorter、OpenSubtitles、OpenWebText2、PhilPapers、PubMed Abstracts、PubMed Central、USPTO Backgrounds、Ubuntu IRC、Youtube Subtitles 从 The Pile (?) 上的各种数据集的验证损失，结果如图 3 、图 8 和图 9 所示。

在第 ?? 节中，我们使用 Huber Loss 来测量域向量的相似性。从技术上讲，Huber loss 结合了 L1 和 L2 距离的优点。在表 2 中，我们添加了不同分布测度的结果。如表 2 所示，Huber Loss 比 L1/L2/JS 距离表现更好。此外，Wasserstein 距离是一个非常优秀的选项。然而，它需要额外的度量空间矩阵 $\boldsymbol{M}$ 来测量两个域向量之间的距离。在这项工作中，度量空间 $\boldsymbol{M} \in \mathbb{R}^{260 \times 260}$ 实际上是每两个元域之间的“数据集转换成本”，且并非容易处理。$\boldsymbol{M}$ 中的每个元素可以通过 $\mathcal{L}_{i,j}$ 来估计，即在元域 j 上训练后在元域 i 处的损失，这需要额外的计算资源。考虑到 Huber Loss 已经取得了非常积极的结果，我们没有进行这个实验。我们相信，Wasserstein 距离也可以表现出积极的结果（甚至更好），如果度量空间被很好地估计的话，我们将其留作未来的工作。

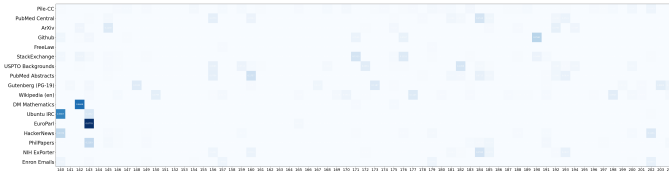

Figure 7. The Pile（?）的每个子数据集的领域向量，其中每一行对应一个子数据集，每一列对应一个元域。数据属于特定元域的比例越高，相应单元格的颜色就越接近 蓝色 。另外，由于 The Pile 主要由英文文本构成，为了清晰起见，我们仅显示英文元域的分布。

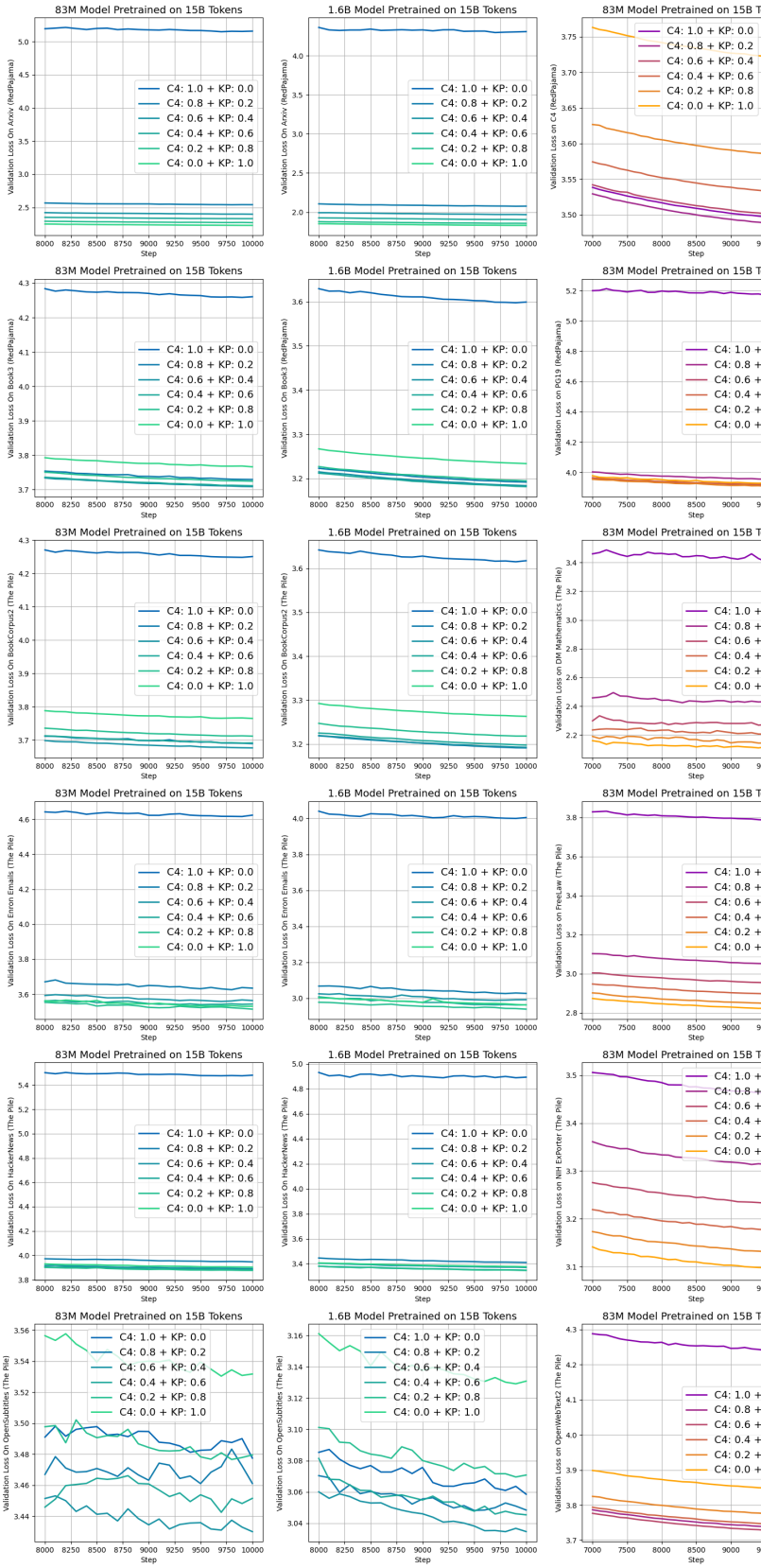

Figure 8. 表格 ?? 中使用数据混合训练的模型在不同数据集上的验证损失。

Table 2. Huber 损失比 L1/L2/JS 距离表现更好。

Distributional Measure	Pearson	Spearman
Huber Loss	0.5833	0.6657
JS Distance	0.4527	0.5000
L1 Distance	0.4830	0.5400
L2 Distance	0.5720	0.6429

Table 3. 来自不同基线的 The Pile（?）的数据混合，与用于 ？的数据混合一致。

Data Mixture	Human	DoReMi	Pile-CC Only	RegMix
ArXiv	0.134	0.004	0.0	0.001
FreeLaw	0.049	0.005	0.0	0.001
NIH ExPorter	0.007	0.008	0.0	0.001
PubMed Central	0.136	0.006	0.0	0.003
Wikipedia (en)	0.117	0.086	0.0	0.016
DM Mathematics	0.025	0.002	0.0	0.0
Github	0.054	0.022	0.0	0.0
PhilPapers	0.003	0.034	0.0	0.0
Stack Exchange	0.118	0.019	0.0	0.0
Enron Emails	0.004	0.009	0.0	0.002
Gutenberg (PG-19)	0.025	0.009	0.0	0.002
Pile-CC	0.142	0.743	1.0	0.87
Ubuntu IRC	0.009	0.011	0.0	0.064
EuroParl	0.005	0.008	0.0	0.0
HackerNews	0.01	0.016	0.0	0.012
PubMed Abstracts	0.107	0.014	0.0	0.024
USPTO Backgrounds	0.053	0.004	0.0	0.002

Table 4. 由 Domain2Vec + DA ^{2} 和 Domain2Vec + RegMix 预测的最佳数据混合。为了避免过拟合问题，Pile（?）的任何子集最多只会训练一个周期。我们采用拒绝采样来过滤掉某些不合理的数据混合。因此，预测的数据混合可能会随着模型规模的变化而改变。

Data Mixture	Domain2Vec+DA ^{2}				Domain2Vec+RegMix			
	106M	290M	595M	1B	106M	290M	595M	1B
ArXiv	0.0131	0.0131	0.0389	0.0431	0.0152	0.0070	0.0114	0.0103
FreeLaw	0.0076	0.0076	0.0316	0.0305	0.0395	0.0267	0.0339	0.0268
NIH ExPorter	0.0008	0.0008	0.0028	0.0023	0.0000	0.0199	0.0000	0.0000
PubMed Central	0.0773	0.0773	0.0519	0.0704	0.0343	0.0576	0.0099	0.0518
Wikipedia (en)	0.2970	0.2970	0.2049	0.2126	0.0847	0.0101	0.1014	0.2577
DM Mathematics	0.0003	0.0003	0.0056	0.0026	0.0177	0.0018	0.0011	0.0008
Github	0.0096	0.0096	0.0290	0.0298	0.0034	0.0538	0.0500	0.0138
PhilPapers	0.0018	0.0018	0.0093	0.0025	0.0118	0.0005	0.0333	0.0401
Stack Exchange	0.0464	0.0464	0.0661	0.0585	0.0698	0.0430	0.1199	0.0262
Enron Emails	0.0000	0.0000	0.0009	0.0000	0.0018	0.0000	0.0000	0.0000
Gutenberg (PG-19)	0.0217	0.0217	0.0484	0.0370	0.0467	0.0223	0.0007	0.0252
Pile-CC	0.4338	0.4338	0.3191	0.3814	0.5370	0.6323	0.5546	0.4704
Ubuntu IRC	0.0022	0.0022	0.0063	0.0072	0.1019	0.0123	0.0161	0.0069
EuroParl	0.0003	0.0003	0.0042	0.0040	0.0070	0.0037	0.0116	0.0000
HackerNews	0.0154	0.0154	0.0521	0.0199	0.0028	0.0551	0.0170	0.0673
PubMed Abstracts	0.0596	0.0596	0.0739	0.0532	0.0259	0.0102	0.0190	0.0017
USPTO Backgrounds	0.0130	0.0130	0.0549	0.0449	0.0004	0.0438	0.0201	0.0010

Table 5. 我们在第 1.3 节和第 ?? 节中使用了不同模型的参数。在计算模型参数时，我们不考虑嵌入层和语言模型头层。

Parameter	Text Generation		Downstream Task			
	83M	1.6B	106M	290M	595M	1B
Hidden Size	768	2,048	768	1,280	1,536	2,048
FFN Hidden Size	2,048	5,504	2,048	3,392	4,096	5,440
Num of Layers	12	24	15	15	21	21
Num of Heads	12	16	12	10	12	32
Max Seq Length	4,096	4,096	4,096	4,096	4,096	4,096
Vocab Size	128,256	128,256	151,936	151,936	151,936	151,936
RoPE Base	10,000	10,000	10,000	10,000	10,000	10,000

Table 6. 在 106M 模型上不同数据组合的下游任务性能。类似于？，Human 指的是来自 The Pile 的原始数据组合。Pile-CC Only 仅指在 Pile-CC 子集上训练。DoReMi 和 RegMix 的数据组合和估计计算量来自？。

Benchmark	Human	DoReMi	Pile-CC Only	RegMix	Domain2Vec + DA [2]	Domain2Vec + RegMix
106M Model Pretrained on 2B Tokens						
Social IQA	0.340	0.349	0.353	0.356	0.339	0.342
HellaSwag	0.268	0.268	0.269	0.269	0.267	0.264
PiQA	0.573	0.584	0.580	0.586	0.579	0.583
OpenBookQA	0.245	0.251	0.249	0.242	0.245	0.249
Lambada	0.065	0.099	0.102	0.091	0.091	0.090
SciQ	0.550	0.520	0.509	0.537	0.549	0.518
ARC Easy	0.329	0.339	0.335	0.337	0.334	0.331
COPA	0.525	0.570	0.572	0.585	0.578	0.557
RACE	0.236	0.254	0.246	0.251	0.240	0.244
LogiQA	0.282	0.280	0.271	0.274	0.268	0.286
WinoGrande	0.516	0.516	0.502	0.508	0.506	0.499
MultiRC	0.539	0.520	0.515	0.533	0.541	0.544
Average Performance	0.372	0.379	0.375	0.381	0.378	0.376
290M Model Pretrained on 6B Tokens						
Social IQA	0.364	0.373	0.374	0.371	0.371	0.368
HellaSwag	0.295	0.312	0.317	0.315	0.307	0.312
PiQA	0.605	0.631	0.639	0.642	0.624	0.633
OpenBookQA	0.261	0.271	0.271	0.262	0.268	0.266
Lambada	0.175	0.208	0.206	0.210	0.182	0.208
SciQ	0.711	0.682	0.663	0.674	0.670	0.697
ARC Easy	0.395	0.410	0.419	0.417	0.420	0.412
COPA	0.632	0.660	0.682	0.657	0.627	0.642
RACE	0.265	0.280	0.280	0.276	0.283	0.281
LogiQA	0.283	0.293	0.296	0.276	0.277	0.292
WinoGrande	0.511	0.506	0.509	0.524	0.498	0.504
MultiRC	0.507	0.555	0.513	0.545	0.521	0.517
Average Performance	0.417	0.432	0.431	0.431	0.421	0.428
595M Model Pretrained on 12B Tokens						
Social IQA	0.378	0.387	0.390	0.394	0.383	0.388
HellaSwag	0.338	0.377	0.386	0.385	0.355	0.366
PiQA	0.624	0.656	0.663	0.667	0.651	0.659
OpenBookQA	0.273	0.279	0.283	0.294	0.288	0.271
Lambada	0.255	0.294	0.332	0.310	0.269	0.292
SciQ	0.777	0.757	0.770	0.791	0.763	0.769
ARC Easy	0.439	0.453	0.478	0.481	0.453	0.460
COPA	0.642	0.680	0.672	0.663	0.668	0.667
RACE	0.289	0.309	0.311	0.311	0.288	0.303
LogiQA	0.263	0.268	0.252	0.267	0.263	0.267
WinoGrande	0.509	0.515	0.506	0.509	0.512	0.503
MultiRC	0.516	0.533	0.522	0.507	0.506	0.527
Average Performance	0.442	0.459	0.464	0.465	0.450	0.456
1B Model Pretrained on 20B Tokens						
Social IQA	0.387	0.411	0.406	0.406	0.394	0.401
HellaSwag	0.375	0.427	0.431	0.436	0.410	0.410
PiQA	0.658	0.684	0.693	0.691	0.684	0.680
OpenBookQA	0.278	0.298	0.300	0.304	0.299	0.302
Lambada	0.301	0.359	0.348	0.353	0.334	0.339
SciQ	0.802	0.822	0.809	0.828	0.821	0.818
ARC Easy	0.482	0.508	0.512	0.518	0.500	0.499
COPA	0.683	0.692	0.713	0.708	0.678	0.698
RACE	0.306	0.319	0.313	0.314	0.305	0.300
LogiQA	0.259	0.258	0.269	0.272	0.268	0.267
WinoGrande	0.513	0.527	0.541	0.512	0.535	0.533
MultiRC	0.523	0.504	0.510	0.530	0.529	0.548
Average Performance	0.464	0.484	0.487	0.489	0.480	0.483
Estimated FLOPs	0	3.7×10^{19} (100%)	0	3.5×10^{18} (9.46%)	9.66×10^{16} (0.26%)	9.66×10^{16} (0.26%)

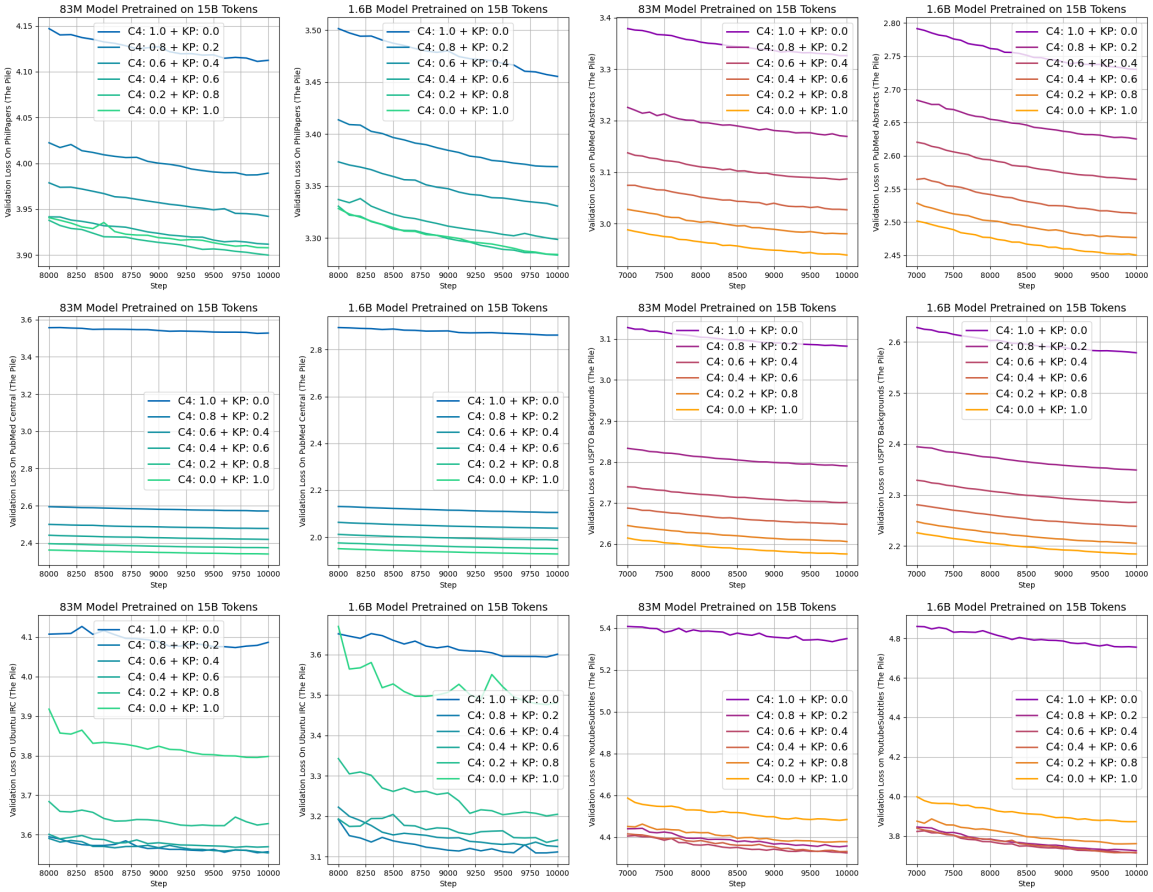

Figure 9. 使用数据混合在表格 ?? 中训练的模型在不同数据集上的验证损失。