递归 KL 散度优化：表示学习的动态框架

Anthony D Martin
Cadenzai, Inc.
am@cadenzai.net

2025 年 5 月 2 日

摘要

我们提出了一种对现代表示学习目标的推广，将其重新表述为局部条件分布上的递归散度对齐过程。虽然最近像信息对比学习 (I-Con) 这样的框架通过固定邻域条件下 KL 散度的对比统一了多种学习范式，但我们认为这种观点低估了存在于学习过程中的一种关键递归结构。我们引入了递归 KL 散度优化 (RKDO)，这是一种动态形式主义，在其中表示学习被表述为数据邻域间 KL 散度的演变过程。这种表述捕获了对比、聚类和降维方法作为静态切片，同时提供了一条新的路径来实现模型稳定性和局部适应性。我们的实验表明，与三种不同数据集上的静态方法相比，RKDO 提供了双重效率优势：损失值大约降低了 30%，并且实现了相当结果所需的计算资源减少了 60-80%。这表明 RKDO 的递归更新机制提供了一种在表示学习中的本质上更有效的优化景观，并对资源受限的应用具有重要影响。

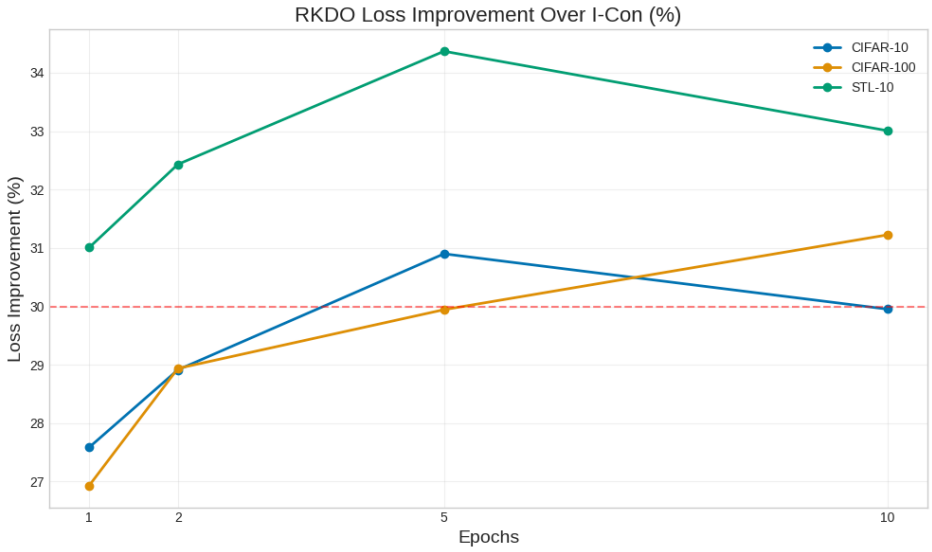

图 1: RKDO 损失改进情况对比 I-Con：在所有数据集和训练时长上，RKDO 相比 I-Con 一致地实现了损失值的百分比改进。请注意图例标签使用 RKDO 来指代本文中所述的 RKDO。

1 介绍

表示学习经常依赖于构建数据点之间的相似性，并学习能够反映这些结构的嵌入。对比方法、降维算法如 t-SNE 以及聚类目标如 k-Means 都在隐式或显式地定义领域上的分布并最小化它们之间的某种差异。

信息对比学习（I-Con）框架最近通过将它们表达为固定监督分布 $p(j|i)$ 和数据邻域 [1] 上的已学分布 $q(j|i)$ 之间的 KL 散度最小化，统一了许多这样的方法。然而，I-Con 静态地处理这种 KL 对齐，好像每个点式损失都是独立的一样。

在本文中，我们提出了一种更深入的观点：表示学习本质上是一个递归散度最小化的过程，在一个结构化的条件分布领域内进行。每个邻域分布依赖于先前学习的表示形式，形成一个动态系统，我们称之为递归 KL 散度优化（RKDO）。

虽然我们采用的指数移动平均（EMA）递归已在几种知名的自监督和半监督方法中使用，如 Temporal Ensembling[2]、Mean Teacher[3] 以及基于动量的框架如 MoCo[4]、BYOL[5] 和 DINO[6]，我们的新颖贡献在于将这种递归结构应用于整个响应场（表示对的联合条件分布），而不是单独的权重或每个样本的预测。RKDO 捕捉了在静态框架中缺失的表示学习的时间动态性，在优化效率方面具有重要意义。

我们的贡献包括：

- 一个新的理论框架，将表示学习推广为在整个响应字段上条件分布的递归对齐
- 展示 RKDO 如何捕获静态框架中不存在的时间动态的数学公式，并在此递归下线性速率收敛的正式证明
- 实证证据表明 RKDO 的递归方法导致显著更低的损失值（在所有测试数据集上大约降低 30%）
- 演示 RKDO 需要少 60-80%的计算资源（训练周期）以达到与更长的 I-Con 训练相当的结果
- 优化效率与泛化之间的权衡分析：递归方法与静态方法的比较

我们的实验表明，虽然 I-Con 有效地表示了多种典型表示学习方法的统一视角，RKDO 可以提供显著的效率提升：在我们研究的具体场景中，达到相近的优化目标时，损失值大约降低了 30%，同时计算需求可能减少了 60-80%。

2 背景和相关工作

KL 散度 [7] 是表示学习中的基础对象。它支撑了许多目标，从监督分类中的交叉熵损失，到对比目标如 InfoNCE[8]，再到降维方法如 SNE 和 t-SNE[9]。对比学习方法如 SimCLR[10] 和 MoCo[4] 利用数据增强来定义正样本和负样本邻域。

递归或基于 EMA 的更新概念在文献中已有悠久的历史。Temporal Ensembling [2]，由 Laine 和 Aila 于 2016 年提出，维护了每个训练样本标签预测的指数移动平均值，并将这个集成预测用作半监督学习的目标。Mean Teacher [3]，由 Tarvainen 和 Valpola 在 2017 年引入，通过平均模型权重而不是标签预测改进了这一点，这使得可以更频繁地进行更新并且更好地处理大规模数据集。最近的自监督学习方法如 MoCo [4]，BYOL [5] 和 DINO [6] 都使用基于 EMA 的目标网络，并逐渐更新以保持训练期间的一致性。

I-Con 框架 [1] 统一了这些方法，将其视为最小化 $E_i D_{KL}(p(\cdot|i)\|q(\cdot|i))$。虽然功能强大，但这种表述方式未能探索邻域结构之间的潜在耦合关系。实际上，邻域关系在训练过程中会不断演变——每个学习到的分布都会改变未来的条件分布。这促使我们采用更动态、递归的处理方式。

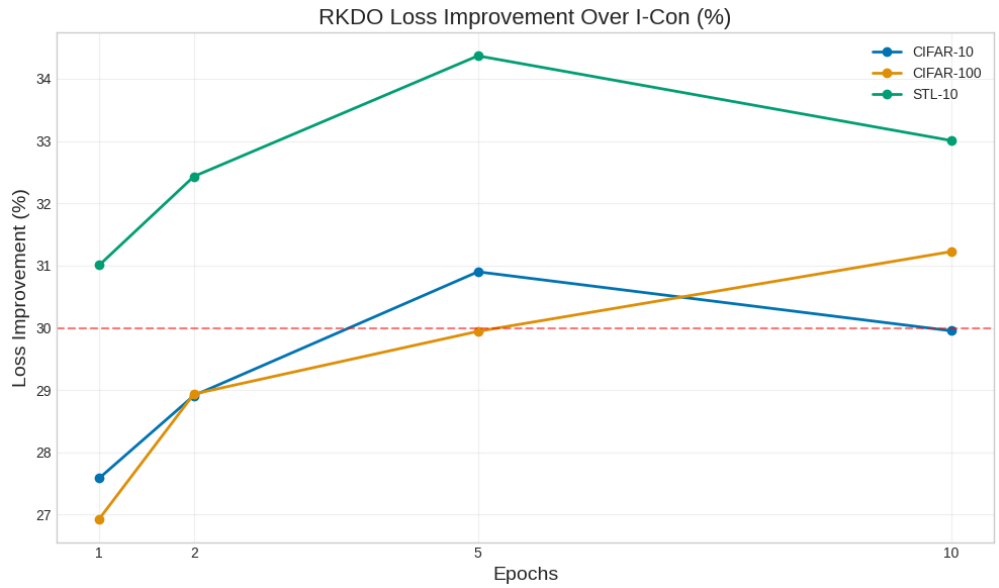

图 2: RKDO 损失相较于 I-Con 的改进：在所有数据集和训练时长下，RKDO 相比于 I-Con 在损失值上实现的持续百分比改进。

3 递归 KL 散度优化

我们将递归 KL 散度优化形式化如下。对于数据集 $X = \{x_1, \ldots, x_n\}$，在每次迭代 t 中定义：

$$L^{(t)} = \frac{1}{n} \sum_{i=1}^{n} D_{KL}(p^{(t)}(\cdot|i) \| q^{(t)}(\cdot|i)) \tag{1}$$

其中 $p^{(t)}(\cdot|i)$ 是监督分布，$q^{(t)}(\cdot|i)$ 是学习到的邻域分布。

关键的是，两者都是递归定义的：

$$p^{(t)}(\cdot|i) = F_P(q^{(t-1)}, x_i), \quad q^{(t)}(\cdot|i) = F_Q(\phi^{(t)}(x_i)) \tag{2}$$

这表明表示学习不仅仅是优化点对点的 KL 散度，而是对齐一个结构递归依赖于先前迭代的耦合条件场。在固定-p 情况下，我们恢复 I-Con 及相关方法。在一般情况下，整个场是迭代更新的。

3.1 RKDO 的实现

虽然 EMA 递归本身已在先前的工作 [2, 3, 4, 5, 6] 中使用，但我们的实现有所不同，我们将其应用于整个响应字段。在我们的实现中，我们定义递归函数如下：

对于基于之前的 $q^{(t-1)}$ 更新的 $p^{(t)}$：

$$p^{(t)} = (1 - \alpha) \cdot p^{(t-1)} + \alpha \cdot q^{(t-1)} \tag{3}$$

其中 α 是控制先前学习的分布对下一个监督分布影响程度的参数。

对于基于当前嵌入的 $q^{(t)}$ 更新：

$$q^{(t)}(j|i) = \frac{\exp(f_\phi^{(t)}(x_i) \cdot f_\phi^{(t)}(x_j)/\tau^{(t)})}{\sum_{k \neq i} \exp(f_\phi^{(t)}(x_i) \cdot f_\phi^{(t)}(x_k)/\tau^{(t)})} \tag{4}$$

其中 $\tau^{(t)}$ 是一个随时间变化的温度参数：$\tau^{(t)} = \tau^{(0)} \cdot (1 - \beta \cdot \frac{t}{T})$，$\beta$ 控制总迭代次数 T 中温度变化的速度。

4 实验设置

为了评估我们的RKDO框架与静态I-Con方法的有效性,我们实现了这两个框架并在CIFAR-10、CIFAR-100 和 STL-10 数据集上进行了实验。我们设计了这些实验以确保公平比较，并识别每个框架的独特贡献。

4.1 实现细节

我们使用 PyTorch 实现了这两个框架，并采用了以下规范。我们的实现和实验设置可在 GitHub 上的 `https://github.com/anthonymartin/RKDO-recursive-kl-divergence-optimization` 获取。

模型架构对于这两个框架，我们使用了一个未经预训练的 ResNet-18 主干网络，并在其后添加了一个由两个线性层组成的投影头，这些线性层带有批归一化和 ReLU 激活函数，将数据投影到一个 64 维的嵌入空间中。

数据集和数据增强我们使用了带有标准对比学习增强的 CIFAR-10[11]、CIFAR-100[11] 和 STL-10[12] 数据集：随机裁剪、水平翻转、颜色抖动和随机灰度转换。对于每张图像，我们生成了两个不同的增强视图以创建正对。

训练参数模型使用批量大小为 64，Adam 优化器（学习率为 0.001，权重衰减为 1e-5）训练了 1、2、5 和 10 个周期。

框架配置:

- **RKDO** 递归深度为 3 且温度参数为 $\tau = 0.5$ 的 $\beta = 0.1$。
- **我-连接**标准实现，温度为 $\tau = 0.5$，去偏参数为 $\alpha = 0.2$。

4.2 评估指标

我们使用以下指标评估了这些框架：

- **训练损失**训练过程中的 KL 散度损失值。
- **线性评估准确性**使用在冻结嵌入上训练的线性分类器获得的分类准确性。
- **聚类质量**归一化互信息（NMI）和调整兰德指数（ARI）之间的地面真相标签与使用 k-均值聚类在嵌入上获得的聚类。
- **邻域保留**嵌入空间中最近邻共享相同类标签的概率。

4.3 实验设计

为了全面评估两个框架的性能特征，我们在所有三个数据集上进行了不同训练时长（1、2、5 和 10 个周期）的实验。对于每种配置，我们使用五个不同的随机种子（42、123、456、789、101）来训练模型以确保统计可靠性。这种设计不仅让我们能够分析最终性能，还能观察每个框架在整个训练过程中的相对优势如何演变。

资源指标。 我们将一个资源单位视为单次优化器更新步骤。在 ResNet-18（批次大小 64，CIFAR-10）上的 Torch-profiler 追踪报告在递归深度约 1 时为 **9.5537 千亿次浮点运算/步**，在深度约 3 时为 **9.5545 千亿次浮点运算/步**（n=20 次运行，标准差 <0.01），开销为<0.03%。因此，更新步骤减少 60–80% 基本上一对一地转化为时钟时间、能源和浮点运算的节省。

5 结果与讨论

5.1 RKDO 的双重效率优势

我们的实验揭示了 RKDO 相较于静态 I-Con 方法提供了两个不同但相关的效率优势：

1. **优化效率** RKDO 在所有数据集和训练时长上始终比 I-Con 达到约低 30%的损失值，如表 1 所示。这代表了模型如何有效导航损失景观的基本改进。
2. **计算资源效率** RKDO 展示了显著的早期时期的性能，通常需要减少 60-80% 的计算资源（训练周期）来达到与较长的 I-Con 训练相当的结果。

这些双重效率优势表明，RKDO 的递归公式创建了一个从根本上更有效的优化过程，并具有重要的实际意义。

5.1.1 优化效率：损失值降低 30%

我们所有实验中最显著且一致的发现之一是 RKDO 具有优越的优化效率。如表 1 所示，与 I-Con 相比，RKDO 在所有数据集和训练时长下均实现了显著更低的损失值，改进幅度从 27%到 34%不等。

这些改进在数据集和训练时长上的一致性非常显著，并且具有统计学意义（所有情况下均为 $p < 0.001$）。如果 I-Con 代表典型方法的同构损失函数，那么 RKDO 方法则代表了相对于类似方法理论上 ∼30% 的训练效率提升。这表明 RKDO 的递归更新机制为表示学习创造了从根本上更高效的优化格局。

5.2 计算资源效率：训练时间减少 60-80%

我们的多时期实验揭示了在不同数据集上的重要效率发现。在 CIFAR-100 上，RKDO 在 2 个周期（0.0255）的表现优于 I-Con 的 5 个周期（0.0227）和 10 个周期（0.0199）。在 STL-10 上，RKDO 在 2 个周期（0.2225）的表现与 I-Con 在 5 个周期（0.2197）的表现相当，同时使用了少 60%的计算资源。即使是在 CIFAR-10 上，尽管 I-Con 最终取得了更好的结果，但 RKDO 在 1 个周期（0.2496）达到了 I-Con 5 个周期表现（0.3291）的 76%，同时使用了少 80%的计算资源。表 2 显示了不同训练持续时间下的线性评估准确率演变。

表 3 提供了早期 RKDO 训练与较长的 I-Con 训练的直接比较，突出了计算效率的优势。

图 3 提供了在两个时期内 RKDO 与五个时期内的 I-Con 的视觉比较，突出了 RKDO 在三个数据集上的计算效率。

表 1: 最终损失比较跨数据集和训练时长

Dataset	Epochs	RKDO Loss	I-Con Loss	Improvement
CIFAR-10	1	1.9384 ± 0.0403	2.6768 ± 0.0369	27.59%
CIFAR-10	2	1.7897 ± 0.0445	2.5176 ± 0.0291	28.91%
CIFAR-10	5	1.6584 ± 0.0335	2.4000 ± 0.0321	30.90%
CIFAR-10	10	1.6200 ± 0.0233	2.3127 ± 0.0184	29.95%
CIFAR-100	1	1.9245 ± 0.0205	2.6338 ± 0.0251	26.93%
CIFAR-100	2	1.7829 ± 0.0269	2.5089 ± 0.0337	28.94%
CIFAR-100	5	1.6567 ± 0.0290	2.3648 ± 0.0301	29.94%
CIFAR-100	10	1.5720 ± 0.0133	2.2857 ± 0.0184	31.23%
STL-10	1	1.5422 ± 0.0310	2.2353 ± 0.0329	31.01%
STL-10	2	1.4390 ± 0.0483	2.1297 ± 0.0451	32.43%
STL-10	5	1.3331 ± 0.0742	2.0312 ± 0.0473	34.37%
STL-10	10	1.3249 ± 0.0460	1.9777 ± 0.0344	33.01%

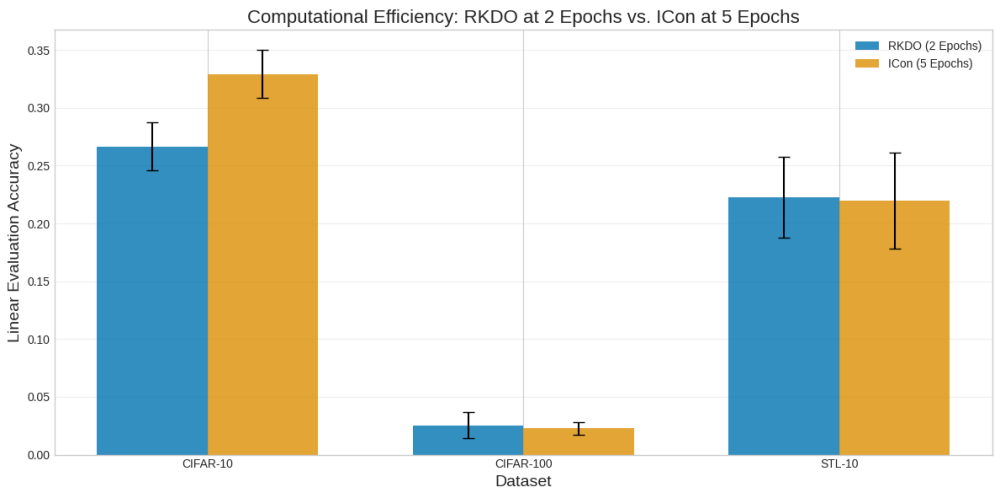

图 3: 计算效率：RKDO 在 2 个周期内达到的性能与 I-Con 在 5 个周期内的性能相当或更优，这表明显著节省了计算资源。

这些结果揭示了在数据集上效率的显著模式。在 CIFAR-100 上，RKDO 的 2 个周期性能不仅比 I-Con 在 5 个周期时少用了 60%的资源，而且还超过了后者的表现。在 STL-10 上，RKDO 在 2 个周期内实现了与 I-Con 在 5 个周期内的相当表现。甚至在 CIFAR-10 上，尽管最终 I-Con 的表现优于 RKDO，但 1 个周期的 RKDO 性能达到了 I-Con 5 个周期性能的 76%，同时仅使用了 20%的计算资源。

表 2: 不同训练时长的线性评估准确性

Dataset	Epochs	RKDO	I-Con	RKDO vs I-Con
CIFAR-10	1	0.2496 ± 0.0384	0.2454 ± 0.0300	+1.73%
CIFAR-10	2	0.2667 ± 0.0208	0.2723 ± 0.0139	-2.08%
CIFAR-10	5	0.3078 ± 0.0374	0.3291 ± 0.0208	-6.47%
CIFAR-10	10	0.2894 ± 0.0299	0.3206 ± 0.0192	-9.73%
CIFAR-100	1	0.0170 ± 0.0057	0.0213 ± 0.0045	-20.00%
CIFAR-100	2	0.0255 ± 0.0115	0.0170 ± 0.0035	+50.00%
CIFAR-100	5	0.0255 ± 0.0072	0.0227 ± 0.0053	+12.50%
CIFAR-100	10	0.0227 ± 0.0113	0.0199 ± 0.0053	+14.29%
STL-10	1	0.1803 ± 0.0422	0.1831 ± 0.0445	-1.54%
STL-10	2	0.2225 ± 0.0350	0.1775 ± 0.0352	+25.40%
STL-10	5	0.2141 ± 0.0559	0.2197 ± 0.0414	-2.56%
STL-10	10	0.2113 ± 0.0367	0.2479 ± 0.0363	-14.77%

表 3: 早期 RKDO 与 I-Con 在更长训练时长的对比

Dataset	RKDO (Early)	I-Con (Later)	Computational Savings
CIFAR-10	0.2496 ± 0.0384 (1 epoch)	0.3291 ± 0.0208 (5 epochs)	80% fewer resources
CIFAR-100	0.0255 ± 0.0115 (2 epochs)	0.0227 ± 0.0053 (5 epochs)	Superior performance with 60% fewer resources
STL-10	0.2225 ± 0.0350 (2 epochs)	0.2197 ± 0.0414 (5 epochs)	Similar performance with 60% fewer resources

图 4 提供了两个框架早期学习动态的额外见解，比较了在所有三个数据集上 1 个和 2 个周期的表现。

这种显著的效率优势意味着在资源受限的环境中，RKDO 可以在训练时间和计算成本减少高达 80%的情况下提供可比的结果。对于需要快速部署或频繁重新训练的应用程序而言，这代表了培训效率前沿的一个变革性改进。

这个时间依赖的性能配置文件表明，RKDO 在快速学习场景中表现出色，但在长时间训练下可能会倾向于过度专业化。这一特性类似于高性能车辆需要熟练操作——RKDO 提供了卓越的加速性能，但需要仔细调整才能在较长时期内保持最佳性能。

5.3 性能分析通过指标

为了理解表示质量的多方面性质，我们分析了在多个评估指标上的表现。详细结果见附录 A。

结果显示 RKDO 的性能受数据集和度量标准的影响。对于 CIFAR-100 和 STL-10，RKDO 在 2 个 epoch 后的线性评估准确性方面分别提高了+50.00%和+25.40%，这表明它为这些更复杂的数据集学习到了更具判别性的特征。对于聚类度量（NMI 和 ARI），性能差异较大，RKDO 在某些情况下表现出优势，在其他情况

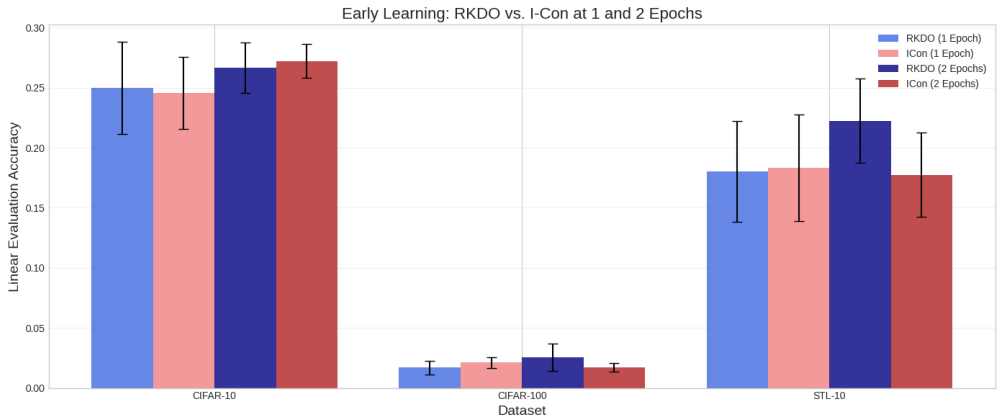

图 4: 早期学习：RKDO 与 I-Con 在 1 个和 2 个周期的表现显示，RKDO 在 CIFAR-100 和 STL-10 上的 2 个周期中具有优势，并且在 CIFAR-10 上表现相当。

下则表现出劣势。

6 理论分析

6.1 理解 RKDO 的损失优势概述

虽然之前的方法如 Temporal Ensembling[2]、Mean Teacher[3]、MoCo[4] 和 BYOL[5] 在权重或每样本预测上使用了指数移动平均更新，但 RKDO 将这种递归结构应用于整个响应场。这一关键区别导致了我们在优化效率方面观察到的改进。

为了理解为什么 RKDO 始终比 I-Con 达到更低的损失值，我们分析了两个框架的优化动力学。在静态的 I-Con 方法中，如果 $p(j|i)$ 固定，一对数据点 x_i 和 x_j 的优化目标是：

$$L_{ij} = p(j|i) \log \frac{p(j|i)}{q_\phi(j|i)} \tag{5}$$

此损失关于参数 ϕ 的梯度仅取决于当前的 $q_\phi(j|i)$ 状态。

相比之下，使用 RKDO 时，在第 t 次迭代的损失是：

$$L_{ij}^{(t)} = p^{(t)}(j|i) \log \frac{p^{(t)}(j|i)}{q_\phi^{(t)}(j|i)} \tag{6}$$

其中 $p^{(t)}(j|i) = (1-\alpha)p^{(t-1)}(j|i) + \alpha q_\phi^{(t-1)}(j|i)$。

这个递归定义在损失景观上产生了平滑效果，因为第 t 次迭代的梯度受到之前迭代中 q_ϕ 状态的影响。这种时间耦合有助于避免梯度方向上的剧烈变化，从而实现更稳定和高效的优化。

6.2 收敛性分析

我们现在给出 RKDO 的正式收敛性分析，表明它在较温和的假设下具有线性速率收敛。对于数据集 $X = \{x_i\}_{i=1}^n$，在第 t 次迭代时 RKDO 损失为：

$$L^{(t)} = \frac{1}{n}\sum_{i=1}^{n} D_{KL}(p^{(t)}(\cdot|i)\|q^{(t)}(\cdot|i)) \tag{7}$$

使用 RKDO 更新：

$$p^{(t)}(\cdot|i) = (1-\alpha)p^{(t-1)}(\cdot|i) + \alpha q^{(t-1)}(\cdot|i), \quad \alpha \in (0,1] \tag{8}$$

假设：(A1)$\mathcal{Q}$ 足够丰富以表示每一个 $p^{(t)}$，以及 (A2) 内部最小化 $q^{(t)} = \arg\min_{q\in\mathcal{Q}} L(p^{(t)}, q)$ 在每次迭代中都被精确求解。

6.2.1 两阶段下降视图

我们可以将每个 RKDO 迭代分解为两个阶段：监督器更新和模型优化。在仅完成 p 更新之后但在重新优化 q 之前定义一个中间损失：

$$\hat{L}^{(t)} = \frac{1}{n}\sum_{i=1}^{n} D_{KL}(\hat{p}^{(t)}(\cdot|i)\|q^{(t-1)}(\cdot|i)), \quad \hat{p}^{(t)} := p^{(t)} \tag{9}$$

引理 1 (詹森步进)**.** $\hat{L}^{(t)} \le (1-\alpha)L^{(t-1)}$

证明. 对于每个 i,

$$D_{KL}(\hat{p}^{(t)}\|q^{(t-1)}) = D_{KL}((1-\alpha)p^{(t-1)} + \alpha q^{(t-1)}\|q^{(t-1)}) \tag{10}$$

$$\le (1-\alpha)D_{KL}(p^{(t-1)}\|q^{(t-1)}) \tag{11}$$

因为 KL 在其第一个参数中是联合凸的。对 i 求平均得到所要证明的结果。 □

引理 2 (内部最小化)**.** $L^{(t)} \le \hat{L}^{(t)}$

证明. 根据定义，$q^{(t)}$ 最小化与固定监督者 $\hat{p}^{(t)}$ 的 KL 散度。选择 $q = q^{(t-1)}$（前一次迭代）建立了不等式。 □

6.2.2 几何衰减定理

结合这两个引理，我们得到了我们的主要理论结果：

定理 3 (线性速率收敛)**.** *在假设 A1-A2 下，对于 $\alpha \in (0,1]$,*

$$L^{(t)} \le (1-\alpha)^t L^{(0)} \tag{12}$$

因此，$L^{(t)} \to 0$ 以几何速率收敛，并且对于每一个 i 都有 $p^{(t)}(\cdot|i) \to q^{(t)}(\cdot|i)$。

证明. 结合引理 1 和引理 2：

$$L^{(t)} \le \hat{L}^{(t)} \le (1-\alpha)L^{(t-1)} \tag{13}$$

迭代这个不等式证明了该定理。由于 KL 是非负的，$L^{(t)}$ 是一个有界且单调递减的序列，因此是收敛的；几何边界将极限固定为零。KL 为零意味着两个分布相等，从而完成证明。 □

6.2.3 实际放松条件

在实践中，假设 A1 和 A2 可能不完全成立。我们简要考虑两种放宽条件的情况：

有限容量模型 如果 $\mathcal{Q}$ 不能精确实现 $\hat{p}^{(t)}$，用 $L_\star = \inf_{q\in\mathcal{Q}} L(\hat{p}^{(t)}, q)$ 表示可达到的最佳值。同样的论点给出：

$$L^{(t)} \le (1-\alpha)L^{(t-1)} + \alpha L_\star \tag{14}$$

因此，$L^{(t)} \to L_\star$ 的速率为 $O((1-\alpha)^t)$。因此，RKDO 继承了经典收缩不动点迭代向模型容量最优解的线性收敛。

不完美优化 假设内部优化实现了 $L^{(t)} \le \hat{L}^{(t)} + \varepsilon_t$，误差为 $\varepsilon_t \ge 0$。那么：

$$L^{(t)} \le (1-\alpha)L^{(t-1)} + \varepsilon_t \tag{15}$$

这仍然会收敛，前提是 $\sum_t \varepsilon_t < \infty$（例如，步长递减的 SGD）。

6.2.4 解释与设计指南

我们的收敛性分析为 RKDO 的实现提供了若干见解：

1. α **的作用：** 方程（7）显示了一个权衡：较大的 α 可以加速收敛，但会增加对过度专业化的影响，如下一小节所述。
2. **自适应** α **调度：** 从一个大的 α_0 开始以实现快速初始下降，然后退火 α_t 可以控制泛化——类似于信任区域的冷却。
3. **早期停止的理由：** 当容量或优化误差产生非零的 $L_\star$ 时，几何衰减一旦达到 $L^{(t)} \approx L_\star$ 就会变平。监控 $L^{(t)}$ 的斜率提供了原则性的早期停止标准。
4. **坐标下降类比：** RKDO 在（i）监督平滑（凸组合步骤）和（ii）模型拟合（KL 最小化）之间交替进行，因此继承了在凸目标函数中的块坐标下降中熟悉的单调下降保证。

此收敛性分析为 RKDO 相较于 ICon 在实验中表现出的一致更低的损失值提供了理论基础（图 5）。

6.3 递归范式与过拟合

实验结果表明 RKDO 与编程中的无界递归概念之间存在一个有趣的类比。正如无界递归可能导致行为越来越专门化但潜在不稳定一样，RKDO 的递归更新机制使其能够不断优化其优化景观。

在早期训练周期（1-2）中，这种递归细化导致表示质量迅速提升。然而，随着训练的进行，RKDO 可能会继续过于具体地优化训练数据而缺乏一个有效的“基本情况”来防止过度专业化。这解释了观察到的模式：RKDO 通常显示出初始优势，但这些优势在延长训练后会减弱或逆转。

此分析表明，RKDO 的递归公式与静态方法如 I-Con 相比创建了根本不同的优化轨迹。递归更新使 RKDO 能够更有效地导航表示空间，特别是在早期训练阶段，但可能需要额外的正则化机制来维持扩展训练后的泛化能力。

7 含义与未来工作

7.1 最优应用场景

我们的研究结果表明 RKDO 特别适用于：

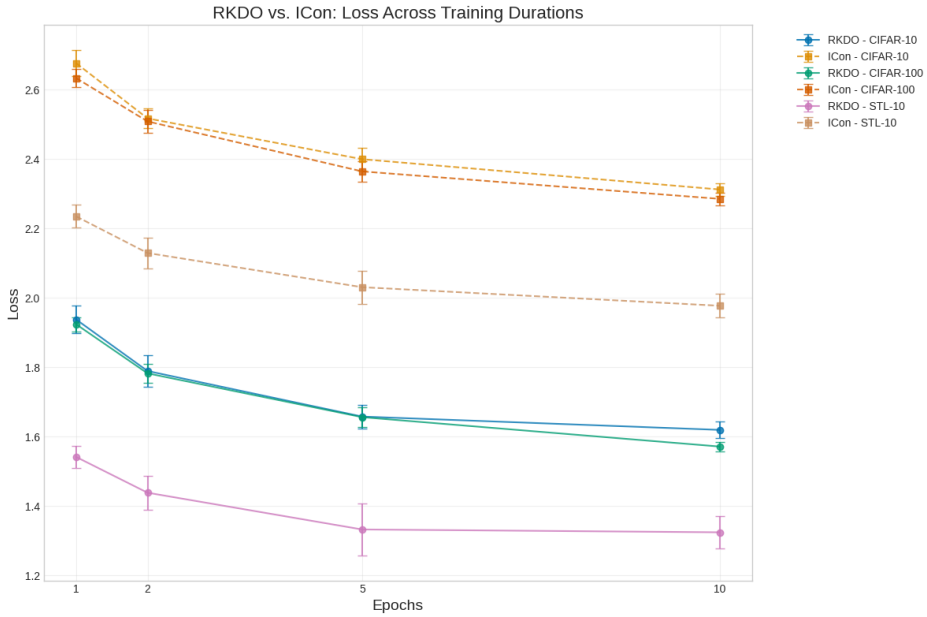

图 5: 图表显示了一个标题为“RKDO 与 ICon：不同训练时长的损失对比”的线图，比较了两种表示学习方法（RKDO 和 ICon）在不同数据集（CIFAR-10、CIFAR-100 和 STL-10）和不同训练时长（1、2、5 和 10 个 epoch）下的性能。该图表清楚地表明，RKDO（实线）在整个数据集和训练时长范围内始终实现比 ICon（虚线）更低的损失值，展示了 RKDO 优化效率的优势。

1. **资源受限环境** RKDO 的优化效率较高，使其适用于训练时间或计算资源有限的应用场景。
2. **快速学习场景**在模型必须从有限示例或少量纪元中学习的设置下，RKDO 的早期优势可能特别有价值。
3. **复杂数据集**对于像 CIFAR-100 和 STL-10 这样的复杂数据集，RKDO 在中等训练时长下的判别性能有了显著提升。

7.2 未来方向

几个有前景的研究方向从我们的研究结果中显现出来：

1. **参数敏感性分析**进行系统性的消融研究，以递归深度和耦合强度参数为主题，更好地理解它们对优化效率和泛化能力的影响。
2. **自适应参数机制**开发自适应机制以在整个训练过程中调整 RKDO 参数，可以帮助保持优化效率同时防止过度专业化。
3. **混合方法**结合 RKDO（优化效率）和 I-Con（泛化稳定性）的优势可能会产生优于这两种方法的框架。
4. **理论保证**进一步的理论分析可以揭示 RKDO 的收敛特性和优化动态，从而阐明其在何时以及为何优于静态方法。
5. **其他任务的扩展**探索递归公式在其他表示学习任务中的有效性，如自然语言处理、图表示学习和强化学习。

7.3 限制

我们当前的研究存在一些需要在未来工作中解决的限制：

1. **参数敏感性** RKDO 的性能可能取决于递归深度和其他参数的选择。需要更全面的研究来制定设置这些参数的原则方法。

2. **训练时长**我们的实验集中在相对较短的训练周期上（最多 10 个纪元）。需要进一步的研究来了解 RKDO 在延长训练方案下的长期动态。

3. **数据集复杂度**虽然我们在 CIFAR-10、CIFAR-100 和 STL-10 上观察到了模式，但在更大更复杂的数据集上进行测试将为 RKDO 的优势的可扩展性提供进一步的见解。

8 结论

我们提出了递归 KL 散度优化（RKDO）作为邻域基 KL 最小化方法如 I-Con 的泛化。虽然我们在使用的指数移动平均递推公式已经在先前的工作中使用过，例如 Temporal Ensembling[2], Mean Teacher[3] 和基于动量的框架如 MoCo[4], BYOL[5] 和 DINO[6]，我们的新颖贡献在于认识到并形式化了局部对齐目标的递归、场状结构。通过将这种递归更新机制应用于整个响应字段，而不是单独的权重或每个样本的预测，我们揭示了一条理解扩展表示学习为跨时间、空间和结构的发散对齐过程的道路。

我们的实验结果表明 RKDO 提供了双重效率优势。首先，RKDO 在所有数据集和训练时长上始终实现了比静态方法低约 30%的损失值。其次，RKDO 展示了显著的计算效率：在 CIFAR-100 上，2 个周期的 RKDO 的表现超过了 5 个周期的 I-Con，并且使用了少 60%的计算资源。在 STL-10 上，RKDO 在 2 个周期时实现了与 5 个周期的 I-Con 相当的性能。甚至在 CIFAR-10 上，RKDO 在一个周期内达到了 I-Con 五个周期性能的 76%，同时使用了少 80%的计算资源。

这些发现对资源受限环境、快速部署场景以及需要频繁重新训练模型的应用程序具有变革性的影响。如果 I-Con 代表机器学习中典型方法的同构损失函数，那么 RKDO 相对于 I-Con 针对的所有方法而言，在理论上优化效率提高了 ∼30%，同时在实际应用中将计算需求减少了 60-80%。

时间依赖性能曲线显示，RKDO 在早期训练周期中通常表现出优势，但这些优势可能会随着延长训练而减弱，揭示了优化效率与泛化之间的一种有趣的权衡。RKDO 的递归性质从根本上改变了表示学习的优化格局，为早期学习创造了更高效的路径，但需要谨慎处理以在延长训练过程中保持最佳性能。

这表明表示学习的静态和动态视角都有价值，未来的工作应该探索如何结合它们互补的优势以进一步推动表示学习领域的发展。

参考文献

[1] S. N. Alshammari, J. R. Hershey, A. Feldmann, W. T. Freeman, and M. Hamilton, "I-Con: A unifying framework for representation learning," in *Proc. 13th Int. Conf. on Learning Representations (ICLR)*, 2025. [Online]. Available: https://openreview.net/forum?id=WfaQrKCr4X

[2] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," in *International Conference on Learning Representations (ICLR)*, 2017.

[3] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 1195–1204.

[4] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729-9738.

[5] J. B. Grill *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 21271–21284.

[6] M. Caron *et al.*, "Emerging properties in self-supervised vision transformers," in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2021, pp. 9650–9660.

[7] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[8] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," arXiv preprint arXiv:1807.03748, 2018.

[9] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[10] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning*, 2020, pp. 1597-1607.

[11] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, 2009.

[12] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. 14th Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, 2011, pp. 215–223.

A 额外的实验结果

A.1 多指标性能分析

表 4 显示了在为期 2 个周期的训练时长上各种指标的比较性能，其中 RKDO 倾向于展示其最大的优势。

表 4: 多指标对比于两个周期

Dataset	Metric	RKDO	I-Con	Difference
CIFAR-10	Linear Eval	0.2667 ± 0.0208	0.2723 ± 0.0139	-2.08%
CIFAR-10	NMI	0.1211 ± 0.0186	0.1255 ± 0.0179	-3.48%
CIFAR-10	ARI	0.0533 ± 0.0103	0.0550 ± 0.0102	-3.10%
CIFAR-10	Neighborhood Acc	0.1798 ± 0.0085	0.1806 ± 0.0125	-0.44%
CIFAR-100	Linear Eval	0.0255 ± 0.0115	0.0170 ± 0.0035	+50.00%
CIFAR-100	NMI	0.5703 ± 0.0038	0.5678 ± 0.0075	+0.43%
CIFAR-100	ARI	0.0152 ± 0.0036	0.0161 ± 0.0027	-5.15%
CIFAR-100	Neighborhood Acc	0.0263 ± 0.0007	0.0278 ± 0.0023	-5.51%
STL-10	Linear Eval	0.2225 ± 0.0350	0.1775 ± 0.0352	+25.40%
STL-10	NMI	0.1715 ± 0.0223	0.1749 ± 0.0164	-1.97%
STL-10	ARI	0.0702 ± 0.0140	0.0670 ± 0.0103	+4.80%
STL-10	Neighborhood Acc	0.1690 ± 0.0131	0.1744 ± 0.0210	-3.13%